.EXE

NOVEMBER 1993 VOL 8 ISSUE 6

£3.20

The Software Developers' Magazine

Hey good-lookin'. Stop that and look at me.
Read our bug-eyed view of C and C++

Speaking of bugs. Can't trust anyone these days:
We reveal why ISO/IEC 9899:1990 isn't immune.

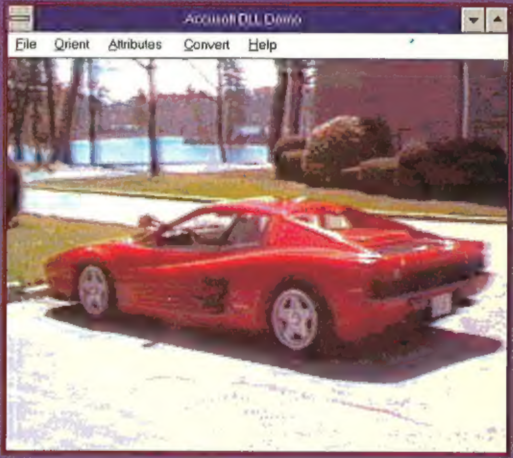
Cheshire cats and abstract classes...

Virtual reality with attitude -
Adding behaviour to VR apps.

Image Format Library

Add TIFF, PCX, DCX, BMP, GIF, DIB, TARGA, WPG, WMF, EPS, PICT and JPEG support to your programs in less than 1 Hour!!

Image Format Library for C/C++/Pascal users provides an easy way to add powerful graphic imaging file format support to any application. Since both DOS & Windows versions are included, your application will work in both environments.



The libraries are designed to be easy to use, fast, and reliable. This simplicity comes from a design that hides all format complexity from the programmer. The libraries are fast because they employ modern optimization techniques- that include assembler code where necessary.

A single function call converts any of the file formats into a standardized format that can be used in any way by your application, even if a TIFF file has a PCX extension. It should take less than an hour to write import and export support for an application, and there will be no format-specific code, thus there is only one function to read and write, and everything defaults to the most commonly supported compression scheme. Compression type may also be specified.

A second-level interface is supported that provides a high degree of control over all functions as well as memory usage, plus the ability to read or write sub-regions, read and display in one step, and grab the screen directly to a compressed file. As the Windows Library is in DLL form, any Windows language can have instant image import/export, display, and printing features. Under DOS, all languages and compilers are supported. Image Format Library supports the following file formats: TIFF, PCX, DCX, BMP, GIF, DIB, TARGA, WPG, WMF, EPS, PICT, JPEG.
The VBX version is compatible with Visual Basic and Visual C++, and all features are implemented as properties of the control, rather than as function calls.

Available as a DLL or VBX

▶ £345 ◀

+ Carriage + VAT

ExpoTech

CORPORATION

The Leaders in Development Tool Distribution

ExpoTech House, Western Road, Hove, Sussex. BN3 1AF.
Sales: +44-(0)273-749222 Fax: +44-(0)273-746446

➤ CIRCLE NO. 122

Vol 8 Issue 6

.EXE

November 1993

Acting Editor: Cliff Saran
Production Manager: Kate Adams
Deputy Production Manager: Mark English
Advertising Manager: Marc Warren
Publisher: Sandra Inniss-Palmer
Printed by: Chase Web Ltd, St Ives plc
Cover Picture: CNRI/Science Photo Library

False-colour scanning electron micrograph (SEM) of the head of the minute buffalo gnat Similium sp. Its poisonous bites and blood-sucking habits cause irritation and sickness in humans.

General

.EXE Magazine is independent and not affiliated to any vendor of hardware, software or services. It is published by Process Communications Ltd, St Giles House,
50 Poland Street, London W1V 4AX.
Tel: (Advertising/Editorial/Production) 071 287 5000
(Subscriptions) 071 287 5000
Facsimile: 071 437 1350 ISSN: 0268-6872

Subscriptions

.EXE Magazine is a monthly journal for software developers. It is available only by subscription, at a cost of £35 per annum (12 issues) in the UK, £56 for 2 years. Overseas subscriptions cost £50 per annum, £70 for 2 years. 'A subscription implies that this journal will be sent to the subscriber until one of the three expires' - AG Macdonell. The magazine is published around the 1st of the month. To subscribe or obtain details of overseas/academic rates, please call 071 287 5000, or write to The Subscriptions Manager, .EXE Magazine, St Giles House, 50 Poland Street, London W1V 4AX. We can invoice your company or take ACCESS and VISA cards. In case of a query about your subscription, please call 071 287 5000.

Back issues are available at a cost of £3.50 each - call our West End office (071 287 5000) for a list of issues still in print.

Editorial

Editorial enquiries should be addressed to The Editor, .EXE Magazine, St Giles House, 50 Poland Street, London W1V 4AX or emailed to **willw@dotexe.demon.co.uk**. We welcome letters, opinions, suggestions and articles from our readers. Please write for a copy of our Contributors' Guide.

Information contained in .EXE is believed to be correct. If errors are found, we will endeavour to publish a clarification in the next available issue.

From time to time, we offer to copy the PC program code described in an article onto diskette for our readers. In this case, please send a blank, formatted disk with a self-addressed, prepaid mailer to the editorial address given above. We can copy both 5.25" and 3.5" disks.

The publishers can accept no liability for any consequences of using software distributed in this way.

Advertising

If you are interested in advertising in this magazine, please write to the Advertising Manager, .EXE Magazine, St Giles House, 50 Poland Street, London W1V 4AX, or call 071 287 5000 for details of our advertising rates.

Pronunciation

.EXE Magazine rhymes with 'not sexy magazine'.

Copyright

Issue Theme: C and C++

Dear Reader, please note our new address:
St Giles House, 50 Poland Street, London W1V 4AX
Tel: 071 287 5000 Fax: 071 4371350

Welcome to Winville

George Heeston, our travel correspondent, gives a guided tour of Windows jargon.

Many programmers who make the transition to Windows say that they find the new environment hostile. They are of course quite right: it is. And it takes time to get used to having messages fired at you whenever your back is turned. You have been transported to Winville, which is populated by many strange beings that are friendly when you treat them right but will bite you if you abuse them. We offer some quick guidelines that may come in handy in the early stages, starting with hint number one: a good memory for names comes in *very* handy.

Your first contact is likely to be with some of the workers. You can tell a worker because he wears a hard hat, almost invariably spells his name in `MixedCaseRunTogether`, and carries a bundle of parameters. Every worker does a very limited job: the demarcation rules are extremely strict. You want to shut a window? You call `CloseWindow` to do it for you (and that is all he does all day - shut windows). A restaurant has separate employees to `SetMenu` and `ModifyMenu`, not to mention the waitress `GetMenuState`. ('Sorry dear, no carets today.') And there are many, many more ...

After the workers, the most conspicuous inhabitants are messengers. Their names are written in `UPPERCASE`, and are `RUNTOGETHER`, except many of them are double-barrelled, or even triple-barrelled, because the clan they come from is important. The most extensive is clan `William`, or `WM_`, which includes well-known figures like `WM_CREATE`, `WM_DESTROY` and `WM_QUIT` along with rare specimens such as `WM_RENDERALLFORMATS` (but, not as it happens, `WM_THECONQUEROR`, or `WM_GATES`). All messengers are issued with Reebock trainers but actually spend most of their time waiting in line. The day might start in a queue at the hardware store. No sooner have they picked up a `click` or a `char` than they are put at the back of another queue, then pushed around from place to place, and all too often end up at the office of `DefWindowProc` who, being `Def`, can't understand what they are telling him and sends them back to the hardware store again. Really, it is surprising they keep so cheerful, especially one like `WM_PAINT`, who hasn't got a priority badge and is always elbowed to the back.

While messengers spend all day running somewhere or waiting to be sent running somewhere else, layabouts are passive and try to do as little as possible. Layabouts have `UPPERCASE` names too, but they often seem designed (like `TTPOLYGONHEADER`, `OLESERVERDOCVTBL` or `GLYPHMETRICS`) to deter the fainthearted from trying to discover what it is they might ever actually be useful. Don't worry; a worker will tell you if you need one and wake him up for you.

Then there are all the drones, which buzz around and get in your hair. They have `UPPERCASENAMES` as well, often double-barrelled, like `R2_MASKNOTPEN`, or maybe `R2_MASKPENNOT`, or would you believe, `R2_NOTMASKPEN`? Remember the hint about a good memory for names. Is it obvious to you which of these is which? Because, if you have to look them up, it would be quicker to type 3 or 5 or 8. To be fair, some drones are jolly useful, but many of them just clutter up the 'namespace' as visitors are told to call it.

There is a lot of talk in Winville about so-called Hungarian naming. Many of the inhabitants emigrated from Europe, where they spoke what was called the strong language', and shouted archaic abuse like 'lpszParam!' at visitors who had lost their way. The best response to this is to reply in kind. Some more recent arrivals have brought in pretty strong stuff of their own, and a good loud 'XST_EXECACKRCVD!' will make them think you know your way around even if you don't.

Traces of more recent history can be found too. There was the very successful seventies pop group the `MetaFile`, who sponsored a DJ called `PlayMetaFileRecord` on the Winville radio button, and the molecular physicists `AddAtom` and `DeleteAtom`, who tried to isolate a universal handle.

Handles are the principal currency in Winville, and many denominations are in circulation; an `HINSTANCE`, an `HWND`, and so on. The coins are all the same size but there is no official way to exchange them. (Experienced Windows programmers keep their handles in a special wallet with twenty labelled compartments.) There are conflicting theories as to whether all the handles in the universe were produced in a primeval big boot or whether they are continuously created by the System. It was hoped that firing a variety of handles from a cyclotron into the Atom Table would show that the `GlobalHandle` was the only one capable of hashing the atom, but (as you will already have guessed) it has a `MixedCase` name and is in fact just a worker posing as a plausible theory.

Perhaps it is not surprising that after two or three weeks spent in Winville some programmers become disoriented. Don't worry, a few years of therapy will sort you out. Come with me on my next visit to `LocalShrink`.

EXE

George Heeston is an alias; do not bother to try to contact her on email.

GREY MATTER

Prigg Meadow, Ashburton
Devon TQ13 7DF

Prices do not include VAT or other local taxes but do include delivery in mainland UK . Please check prices at time of order as ads are prepared some weeks before publication. This page lists some products – call us for a complete price list. ORDER BY PHONE WITH YOUR CREDIT CARD

(0364) 654100

FAX: (0364) 654200

News & Views

The new WindowsMAKER Professional 5.0 from Blue Sky looks like setting new standards for Visual Programming under Windows. Double click on a screen object and add code immediately, like Visual Basic. Supports Resource Workshop & AppStudio, integrates into the Visual C++ Visual Workbench, supports 3rd party editors, etc. Code Generation modules for ANSI C, MFC 2.0 & OWL, generates code for Windows 3.1, Win32s & Windows NT. Not cheap at £875, but should be of interest to any C or C++ programmer writing Windows code for a living. Call us for full details. The Visual Programmer component of Symantec C++ 6.0 is also written by Blue Sky, and includes some of these features. The Standard Edition is only £85 so why not try it out?

The new V5.0 of OptLink for Windows can now generate compressed .EXE and .DLL files. Why not switch to the same linker used to develop the likes of Lotus 123, Word Perfect for Windows, Corel Draw, Word for Windows, Excel and WindowsMAKER.

C & C++ FOR DOS

Comms

Asynch Prof for C/C++	£130
C Asynch Manager	£145
C Comms Toolkit	£215
COMM-DRV	£145
Essential Comms	£215
Greenleaf CommLib 4.0	£225
SilverComm "C" Asynch 4.0	£135

Database

Btree Filer for C Network	£170
Btrieve	£365
C-Index+ 4.0	£250
c-tree Plus 6.0C	£565
CodeBase 5.0	£240
CodeBase++ 5.0	£240
D-ISAM	£325
d-tree	£490
DataBoss	£520
Greenleaf Database Library	£180
Paradox Engine & DFX 3.0	£125
POET SDK 2.0	£360
r-tree	£295
Raima Database Manager	£305
Raima Object Manager	£305
SoftFocus Btree/ISAM	£75
Sycero.C Net	£895
Topaz for C 4.0	£215
Watcom SQL Developer's	£260

Graphics & GUI

Accusoft Image Format Lib	£360
dGE 5.0	£205
Essential Graphics Chart	£130
Essential Graphics Kernel	£135
Essential Graphics GUI	£99
GraphiC 6.1	£305
graphics-MENU & Data Entry	£205
GX Graphics 2.0	£155
HALO Professional 2.0	£245
MetaWINDOW-DOS 4.3	£150
MetaWINDOW-DOS/286 4.3	£210
MetaWINDOW-DOS/386 4.3	£270
object-Menu	£305
PCX Toolkit 5.0	£155
PCXLab-DOS/286/386	£150
Real-Time Graphics & M/CT	£185
T-Windows	£240
TEGL Windows Toolkit 3.0	£180
zApp for DOS Graphics 2.0	£330
Zinc 3.5 for DOS	£354

Math & Scientific

C/Math Toolchest & Grafix	£45
Huge Virtual Array & NAT	£215
Linpack.h++ w/Source	£429
Math.h++ w/Source	£257
MIRACL	£39
Science & Eng Tools 8.0	£115

Printer

Baby Driver Kernel	£185
BGI Printer Driver Toolkit	£105
GX Printer	£240
SLATE	£260
TesSeRact Printer Control Sys	£45

Screen

C-Scape 4.0	£360
C/Windows Toolchest	£45
Greenleaf Datawindows 3.0	£225
MEWEL Window System 4.0	£380
Object Professional for C++	£165
Panel Plus II	£260
TCXL-DOS 6.1	£79
Turbo Vision Dev Toolkit 2.0	£105
Vermont Views Plus	£595
zApp for DOS Text	£330

General & Systems Libraries

C Tools Plus/6.0	£115
CAD/CAM Dev Kit/2D Personal	£375
CodeRunneR	£150
Heap Expander 3.0	£130
Hold Everything	£130
Interwork	£205
MoreHeap	£110
MTASK	£215
Multi-C w/Source	£39
PDK #1, #2, #3	£95
PowerSTOR	£99
Resident-C	£160
TesSeRact Compression Sys	£45
TesSeRact TSR Dev Sys 2.0	£79
TG Professional (CAD/CAM)	£395
Tools.h++ w/Source	£257
TSRs and More	£115
VMData	£370

Tools

C-DOC 5.0	£150
C-Vision 3.1	£90
Clear+ for C	£135
CodeCheck Personal	£250
MKS LEX & YACC	£240
PC-Lint	£85
PCYACC Personal	£250
SourcePrint+ Deluxe 5.0	£190

DOS EXTENDERS

NEW PHAR LAP TNT DOS-EXTENDER SDK 6.0 now lets users of Visual C++ 32-bit Edition build 32-bit DOS applications which take advantage of powerful Windows NT features such as DLL's, threads & multi-tasking. It also includes a 32-bit version of CodeView. ***ONLY £340.***

Phar Lap 286IDOS-Ext SDK	£330
Phar Lap TNT DOS-Ext SDK	£340

C & C++ COMPILERS

NEW SYMANTEC C++ 6.0 PROFESSIONAL now has a Windows-hosted IDE, full AT&T 3.0 C++, targets 16 & 32-bit DOS, Windows & Win32s, royalty-free DOS extender, new Visual Programmer by Blue Sky is more visual than Visual Basic, MFC 2, full drag'n'drop, direct support for PVCS from IDE, OPTLink/Windows, CD-ROM with online documentation, and lots more! SPECIAL OFFER - ***ONLY £149*** *(NORMAL PRICE WILL BE £330). STANDARD EDITION (16-BIT, NO CD)* ***ONLY £85.***

Aztec Embedded C86	£290
Borland C++ 3.1	£250
Borland C++ & AFX 3.1	£365
Microsoft Visual C++ Std	£96
Microsoft Visual C++ Prof	£230
MS QuickC 2.5	£52
MS QuickC & QuickAsm	£87
Symantec C++ 6.0 Std	£85
TopSpeed C Prof	£180
TopSpeed C/C++ Prof	£225
Turbo C++ 3.0	£59
Turbo C++ Visual Ed. for Win	£84
Watcom C/C++16 9.5	£325
Borland C++ for OS/2	£250
High C/C++ 386 3.1	£555
MS Visual C++ 32-bit Edition	£300
NDP C/C++ 4.2 (386)	£445
Symantec C++ 6.0 Prof	£149
Watcom C32 for DOS 9.5	£135
Watcom C/C++32 9.5	£375

VISUAL BASIC FOR WINDOWS

Visual Basic 3.0 for Win Std	£96
Visual Basic 3.0 for Win Prof	£230
Comms Lib/Win	£115
Network Library/Win	£80
PDQ Comm for Windows	£99
Agility/VB	£160
Btrieve for Windows	£365
CodeBasic	£150
ObjecTrieve Prof 2.01	£410
Q+E Database/VB	£99
Q+E Multilink.VB	£295
Q+E ODBC Pack	£140
Raima Data Manager	£380
VB/ISAM MU	£150
VBtrv Control Pack	£140
Visual Controls for Btrieve	£180
Visual/db Network 3.0	£235
3-D Gizmos	£80
ButtonTool & EditTool	£69
ChartBuilder for VB	£75
Control Palette 2.0	£110
Custom Control Factory	£49
MS Visual Control Pack	£69
QuickPak Pro for Win 3.0	£145
Spread/VBX	£205
SpyWorks-VB	£120
VB/Magic Controls	£120
VBAssist 3.0	£135
VBTools/Win 3.0	£145
3D Graphic Tools	£130
Compression Plus for VBWin	£200
GUI Assist	£350
ImageKnife/VBX	£145
Microhelp Muscle/Win	£160
PCX Toolkit for Windows	£155
QuickPak Scientific for Win	£105
Spellcheck/VB	£49
TrueGrid	£80
VB Compress Pro	£95
VBXRef Utility	£80

C & C++ FOR WINDOWS

C Comms Toolkit/Windows	£270
COMM-DRV	£145
CrystalComm for Windows	£130
Greenleaf Comm++	£180
Greenleaf CommLib Pro	£355
Agility/VC++	£160
Btrieve for Windows	£365
CB/ISAM MU for Win	£150
CodeBase 5.0	£240
CodeBase++ 5.0	£240
Integra VDB	£399
Paradox Engine & DFX 3.0	£125
POET SDK 2.0	£360
Q+E Database Library	£295
Raima Database Manager	£380
Raima Object Manager	£305
Watcom SQL for Windows	£260
Essential Graphics Chart	£300
GraphiC/Win	£370
Graphics Server SDK	£205
Windows Charting Tools	£300
Accusoft Image Format Lib	£360
Control Palette 2.0	£110
Data Entry Workshop	£130
DataTable	£195
Diamond Toolkit	£310
Magic Fields	£300
MS Visual Control Pack	£69
Out of Controls w/Source	£355
PCX Toolkit for Windows	£155
ProtoGen+	£295
Toolbox for Windows 3.0	£305
Win/Sys Library	£105
WindowsMaker Pro 5.0	£875
zApp for Windows 2.0	£330
zApp for Windows NT 2.0	£330
Zinc 3.5 for Windows	£354
C++ Designer	£195

PROGRAMMING tools

Ada
Basic
Comms
Custom Controls
Debuggers
Editors
Graphics
Linkers/Locaters
Modula-2
Pascal
Rexx
Smalltalk
Version Control
Windows

Assemblers
C/C++
Cross Dev
Database
Dos Extenders
Fortran
GUI
Lisp
Multi-tasking
Prolog
Scheme
SQL
Visual Programming
Xbase

We stock many items for which there is no space in these advertisements.

LOW PRICES FOR MICROSOFT & BORLAND PRODUCTS

Microsoft Access 1.1	**£285**
Microsoft Delta	**£230**
Microsoft Fortran 5.1	**£96**
Microsoft Fortran Power Station	**£230**
Microsoft FoxPro 2.5 DOS	**£290**
Microsoft FoxPro 2.5 Win	**£290**
Microsoft Macro Assembler 6.1	**£96**
MS Multimedia Viewer Pub Kit	**£230**
MS Visual Basic for DOS Std	**£96**
MS Visual Basic for DOS Prof	**£230**
MS Visual Basic for Win Std	**£96**
MS Visual Basic for Win Prof	**£230**
Microsoft Visual C++ Std	**£96**
Microsoft Visual C++ Prof	**£230**
Microsoft Visual C++ 32-bit	**£300**
Microsoft Windows 3.1	**£72**
MS Windows SDK 3.1	**£172**
Borland C++ 3.1	**£250**
Borland C++ & AFX 3.1	**£365**
Borland Pascal with Objects 7.0	**£250**
dBase IV 2.0	**£415**
dBase Compiler	**£250**
Paradox Engine & DFX 3.0	**£125**
Turbo C++ 3.0	**£59**
Turbo C++ Visual Ed. for Win	**£84**
Turbo Pascal 7.0	**£84**
Turbo Pascal for Windows 1.5	**£99**
Paradox 4.0	**£415**
Paradox for Windows	**£335**

WITH FULL TECHNICAL SUPPORT

VISUAL BASIC FOR DOS

MS Visual Basic for DOS Std	£96
MS Visual Basic for DOS Prof	£230
Comms Lib/DOS	£115
Network Library/DOS	£80
PDQ Comm	£65
Btrieve 5.1	£365
db/Lib Prof 3.0	£195
E-Tree Plus	£165
Elltech's GUI Toolkit	£125
Graphics QuickScreen	£105
Graphics Workshop	£105
GraphPak Pro	£99
GX Graphics 2.0	£155
Compression Plus	£130
Compression Workshop	£110
Microhelp Muscle/DOS	£155
PDQ	£110
Printer Plus	£185
ProBas 5.5	£165
QuickPak Prof 4.1	£145
QuickPak Scientific 3.0	£105
QuickScreen 4.0	£105
Science & Eng Tools 8.0	£80
TSRific	£80

BASIC

Compilers & Interpreters

BBC BASIC-86 Plus	£75
CA-Realizer (Win)	£190
GFA-BASIC for DOS	£85
GFA-BASIC for Windows	£170
GFA-BASIC for Win Compiler	£42
PowerBASIC 3.0	£110
TrueBASIC 3.0	£75

ASSEMBLERS

MS MASM 6.1	£96
Optasm	£195
Phar Lap 386IDOS Ext SDK	£340
Phar Lap 386IASM/Linkloc	£855

Compression Lib

GreenLeaf has announced a data compression library for C and C++ programmers. ArchiveLib comprises over 100 functions including a variety of functions to add, replace, delete, update and retrieve objects of compressed data within the archive. Not only will it handle files and directories, but ArchiveLib also enables programmers to store buffers directly into archives without have to go through an intermediate file stage. The UK price is £229 from Citadel on 0566 86037.

Irish VB users

iVBUG (Irish Visual Basic User Group) will be launched on Wednesday 3rd November 1993 at the Burlington Hotel in Dublin. The group will meet monthly, providing tips, technical support and programming tricks. Each meeting will feature a guest speaker. There will be a bimonthly news-letter called .SetFocus and regular VB workshop sessions. A utility disk containing DLLs, custom controls, sample code and bug lists will be distributed bimonthly to members. Further information is available from Declan Ward on 01035 2988299.

TCP/IP for VBW

NetManage Inc has produced a TCP/IP development kit for VBW which enables VBW developers to access the company's NEWT TCP/IP stack for Windows. The kit is priced at £407. Existing users can upgrade for £100. The UK distributor is Integralis (0734 306060).

Full Motif

Hypersoft, the UK distributor for Rogue Wave Software, has announced a new version of View.h++ which features full support for OSF/Motif 1.2 and X11R5. This means it provides the complete Motif look-and-feel including drag-and-drop, tear-off menus, frame constraints, wide character string functions and updated widgets. View.h++ offers a two-tier hierarchy. the lower-level encapsulates the entire Motif GUI in a class library. The upper-level provides high-level functions which, for instance, enable developers to create dialog boxes using a single API call. View.h++ costs £684 from Hypersoft on 0273 834596.

New NU-Mega

There is a new version of Nu-Mega's Bound-Checker for Windows which now offers an event logging and viewing capability. This provides a way for developers to trace the flow of events preceding a given problem. The most recent events are stored in memory. However events can also be logged to file: Nu-Mega's TVIEW utility can then be used to view the activity of the program. This provides tree-like displays of event hierarchies. Events which are passed to default Windows handler can optionally be ignored by TVIEW. Bound-Checker costs £195 from System Science (071 833 1022).

ObjectWorld '93

Over 1,000 people visited the first Object World UK conference and exhibition at the Heathrow Ramada hotel earlier this month.

The conference, which is organised under the auspices of the Object Management Group industry standards body, followed the pattern set down by its US counterparts of combining seminars on the business and technical implications of object oriented technology.

The exhibition, which ran alongside the conference, was attended by 42 suppliers, although surprisingly few chose to make product announcements at the show itself. For example, ICL, which had splashed out on one of the biggest stands, chose to make its long awaited DAIS object request broker announcement a week earlier.

Of those who did use the conference as a platform to launch their products, Hitachi Europe made the most noise. It lifted the lid on the latest addition to its ObjectIQ family of development tools - Distributed ObjectIQ - which is designed for building UNIX-based client server systems.

Distributed ObjectIQ is intended to simplify programming UNIX interprocess communications. Programmers normally have to look at a server's API to find out how to drive it. C programs are written in order to carry out remote procedure calls. The application program is then interfaced with the C programs.

The new tool takes a server's API specification and builds an interface directly within the application code. For building servers, Distributed ObjectIQ automatically creates the API specification from the object in the server program.

The product is built on top of the Open Software Foundation's Distributed Computing Environment, although Hitachi has pledged support for the OMG's Common Object Request Broker Architecture. However, this will not occur until the second version of the CORBA is ratified, which is unlikely to be before 1995.

Distributed ObjectIQ is planned for beta release during the first quarter of 1994 with full product shipment scheduled for the third quarter. It will be sold as an add-on to ObjectIQ at an expected price of £3,500.

SAS Institute announced that when release 6.09 of the SAS System ships in the autumn it will include object oriented application development techniques on its tick-list of features.These will be enabled by two new facilities - the Metabase and the Object Manager.

The Metabase is an 'intelligent data dictionary that guarantees that the information presented to the user is dynamically updated without requiring the application to be modified,' according to marketing manager Steve Darbyshire. The Object Manager is a set of predefined objects and tools for creating new object classes.

Liant Software announced a new release of its C++/Views application framework to speed up the creation and porting of GUI applications among Microsoft Windows, OS/2 Presentation Manager, OSF/Motif and Apple Macintosh environments. C++/Views V3.0 automatically adjusts the geometry of GUI objects so that when an applications is moved to other platforms, objects stay in their correct proportion and location.

The product will ship for MS-Windows this month. OS/2, Motif, Macintosh and DOS character versions will be available in Q4 1993.

US supplier ParcPlace Systems introduced ObjectBuilder V2.0, which it claims is 'the only GUI builder designed specifically to enhance productivity in UNIX C++ development.' The product supports both Motif and Open Look user interfaces.

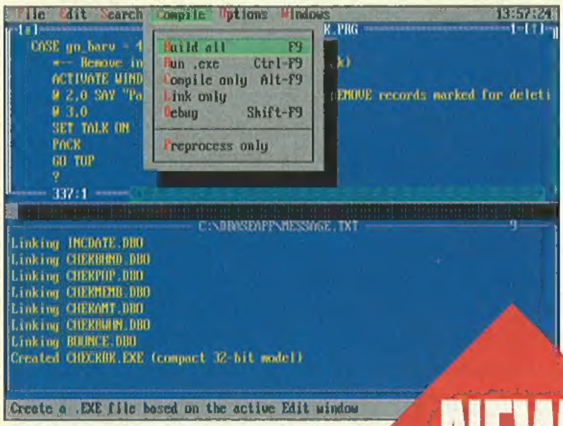
THE WORLDWIDE STANDARD

New 32-bit dBASE Compiler

Create faster applications, faster.

100% compatible, 200% faster

Introducing the new super-fast dBASE® Compiler v2.0 – the only compiler that is 100% compatible with the dBASE language. That means you can compile and run your dBASE III PLUS and IV applications without modification! Not only are your applications 100% compatible, with new 32-bit support, they'll run up to 200% faster than with the dBASE Compiler v1.0!



```
K.PRG
CASE go_barv = 4
   *-- Remove in                              k)
   ACTIVATE WIND
   @ 2,0 SAY "Pa                              REMOVE records marked for deleti
   @ 3,0
   SET TALK ON
   PACK
   GO TOP
   ?
   337:1
```

```
C:\DBASEAPP\MESSAGE.TXT
Linking INCDATE.DBO
Linking CHEKBHND.DBO
Linking CHEKPUP.DBO
Linking CHEKMEMB.DBO
Linking CHEKAMT.DBO
Linking CHEKBWHM.DBO
Linking BOUNCE.DBO
Created CHECKBK.EXE (compact 32-bit model)
```

```
Create a .EXE file based on the active Edit window
```

All command line options are available through our powerful, yet easy to use, menu-driven interface.

NEW!

Build applications faster too!

You'll create applications faster too with v2.0's powerful and intuitive new user interface. It's so easy, you can begin compiling applications instantly. Plus, with the new AutoCompile and AutoLink options, you simply specify the main program and all related files compile and link automatically. Add to that a powerful C-like Preprocessor, MAKE facility, dSPLIT and dJOIN utilities and you'll have the fastest way to build and distribute your dBASE applications available.

Distribute tamper-proof applications royalty-free

Source code security, multi-level password protection and data encryption support from dBASE IV keep all your applications safe. Best of all, with new dBASE Compiler you can distribute applications without any royalties or runtime fees.

So develop faster applications, faster, with new dBASE Compiler v2.0 and move up to a new level of productivity.

Borland
Power made easy

See your dealer today!

What's new in v2.0

- **Create 32-bit .EXEs** that run **200% faster**.
- Put all options at your fingertips with the **new menu-driven interface**.
- Compile and link automatically with **new AutoCompile** and **AutoLink** options.
- Link all your programs, queries, forms reports and label objects in a single file with the **.DBO option**.
- Up to **50% smaller .EXE size**.
- Use **save default option** to store commonly used settings. Plus you'll get **alternative date format, command line wildcard support** and a handy **.MAP file option**.
- And there's much more!

For a free information pack
call 0800 212727
or complete the coupon.

Please send me the FREE information pack.
Ref: AS/9344/DO

Name________________________

Position______________________

Company______________________

Address______________________

____

____

______________________________ .EXE 11/93

Postcode________ Tel.____________

Please return to: Borland International (UK) Ltd, FREEPOST, PO Box 9, Stroud, Glos., GL6 8BR.

➤ CIRCLE NO. 124

Borland® products are available from:

Caerphilly	Software Paradise	0222 887521
Cambridge	Evesham Micros	0223 323898
Cheshire	Inmac	0928 579345
Cheshire	Technology plc	0925 821646
Devon	Grey Matter	0364 654100
Glasgow	Abtex	041 204 4664
Glasgow	Buchanan Ltd.	041 945 5559
Hampshire	Novatech	0705 662299
Herts	Compel	0707 373535
Lancs	P&P	0706 217744
London	Action Computers	0800 333 333
London	MSL	071 731 2566
Manchester	Computer Group	061 872 4363
Middlesex	Technomatic	081 205 9558
Yorkshire	Software Corporation	0904 691391

Borland International (UK) Ltd, 8 Pavilions, Ruscombe Business Park, Twyford, Berkshire RG10 9NN Tel 0734 321150. All Borland product names are trademarks of Borland International, Inc. Dealer prices may vary.

A plus for ProtoGen

ProtoGen+ the long awaited replacement for ProtoGen V3.0 has finally arrived. It features an enhanced screen manager, menu designer, DLL library, custom controls and a dialog editor. There is now complete support for MDI. Other enhancements include range validation, choice checking, DDE link field editing, tool bar support, functions for adding status line help, new fonts and additional 3D effects. An interactive test-mode is provided for assessing the GUI before generation of code. ProtoGen costs £299. Contemporary Software (0273 483979) is offering it at a special promotional price of £149.

Solaris SDK/DDK

A Solaris software developer's kit (SDK) and driver developer kit (DDK) has been made available by SunSoft. The SDK provides the Developer AnswerBook, an online hypertext technical document which contains manuals on ToolTalk, Streams, threads programming and XView. The SDK also includes sample code demonstrating features of Solaris and the OpenWindows Developer's Guide. The DDK provides documentation and sample source for creating Solaris device drivers. The SDK costs £495. An option pack containing C, C++ and Fortran compilers is priced at £1995. The DDK costs £195. SunSoft is on 0494 472900.

Micro Focus/CSI

UK Cobol specialist Micro Focus has bought the rights to the middleware component of its integrated client-server offering. Micro Focus paid an undisclosed sum to acquire the Application-to-Application Interface technology from Creative Systems Interface. The deal gives Micro Focus exclusive use of the technology. CSI's core development and support personnel have joined Micro Focus.

Speak to Me

Voice Assist is a new, speaker-independent, speech recognition software from Creative Labs which includes the VPro command engine from Voice processing Corporation. One way Creative Lab has suggested it could be used is as tool for setting up voice-activated Windows commands. An API is also available. Voice Assist costs £69 from Creative Labs (0743 248590).

Neuron Data licences

Palo Alto-based Neuron Data has decided to drop run-time fees for its complete range of application development tools, from GUIs and data access tools through to its expert systems development products. CEO Patrick Perez said the change was intended to 'eliminate cost and overhead barriers as well as strengthen our service and support.' He claimed the new arrangement would allow developers to distribute applications at no cost, without having to track and manage licences. It will also enable the company to pitch more effectively to large corporates in search of a standard development tool. Such target customers have been put off by the need to manage run-time licences. Under the new scheme, Neuron Data will sell an unlimited deployment per-platform kit for each of the tools in its portfolio. Applications can be redeployed on other platforms by purchasing a porting library, which carries a once-only charge of between £4,000 and £6,000.

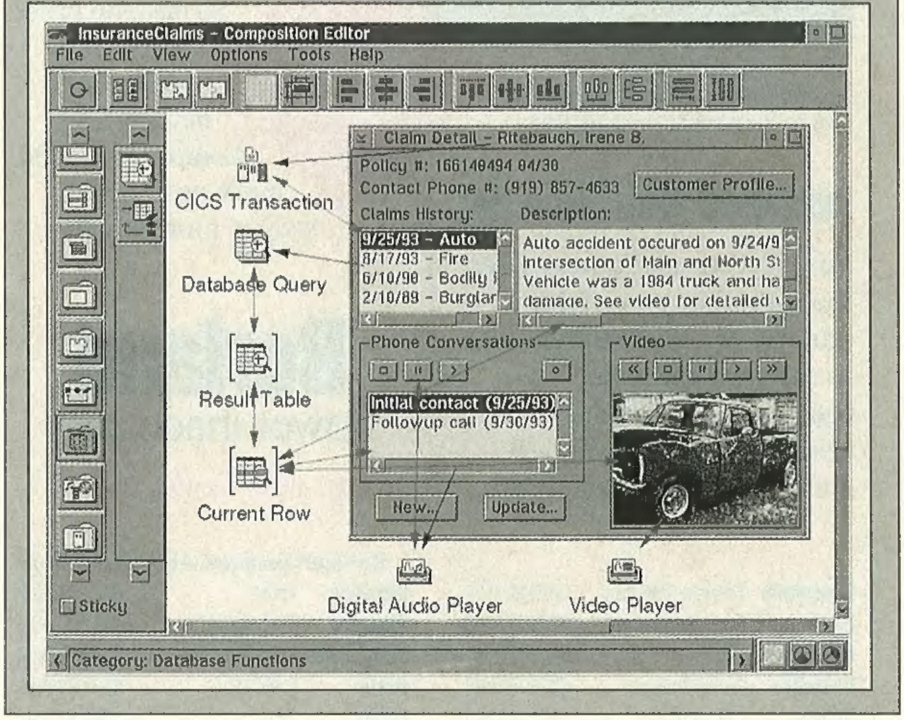
IBM makes Visual Smalltalk

Microsoft and Borland aren't the only companies who name their products 'Visual Something'. VisualAge is IBM's new rapid development tool for OS/2 which is based on Smalltalk. It employs a drag-and-drop metaphor for connecting pre-fabricated software components called *Parts* to build client-server applications. There are Parts which provide CUA-compliant GUIs, Parts for client-server and communications (eg TCP/IP, NetBIOS, APPC and EHLLAPI) and support for both IBM and non-IBM relational databases including DB2/2.

As with other visual design tools, the developer 'draws' a user interface directly on the screen. However, unlike most, the tools for building the GUI are in fact Parts: so buttons, edit field etc map directly onto Smalltalk code. Without having to get down to any actual coding, VisualAge provides the developer with a way to link these controls to methods provided by the Parts. For instance a database Part would offer a browse facility and the ability to insert, update or delete records. No extra work is required by the programmer to attach buttons to these buttons. If extra functionality is necessary, it can be added by delving into Smalltalk.

While the application is being built, VisualAge generates Smalltalk code. A Smalltalk compiler is provided for producing native Presentation Manager applications.

There are two versions of the product: a single-user edition ($2,500) and VisualAge Team ($5,000), a multi-user development system which provides version, change and release control and concurrent access to classes and Parts over a LAN. The product should ship in the first-quarter of '94.



Policy #: 166140494 04/30

Contact Phone #: (919) 857-4633

Claims History:

9/25/93 - Auto
8/17/93 - Fire
6/10/90 - Bodily I
2/10/89 - Burglar

Description:

Auto accident occured on 9/24/9
intersection of Main and North St
Vehicle was a 1984 truck and ha
damage. See video for detailed

Initial contact (9/25/93)
Followup call (9/30/93)

Category: Database Functions

SoftTools for the Job

Systemstar SoftTools has recently been appointed UK distributor for all products from ImageSoft - "The World's Leading Publisher of Software Development Tools."

NEW - Object/Developer

Object/Developer 2.0 is a software development tool created to enable software developers to design, produce and maintain C++ applications. The core environment is a class browser. Object/ Developer provides a compiler independent environment for those who rely on multiple compilers for production and testing of applications. Supports all standard compilers.
Price: Special Introductory - £230

Object/Designer

Create Windows applications as if your job depends on it!

Out performing Visual C++, WindowsMaker and Case W, **Object/ Designer** enables developers to quickly generate all application source files needed for a Windows application. Includes support for zApp 2.0, Microsoft C++/MFC, Borland C++/Object Windows, Turbo Pascal for Windows and CommonView with instant access to third party editors. The features of **Object/ Designer** from Image/Soft make it the only code generation tool flexible enough to be used to write full function Windows applications
Price: £370

ObjecTrieve/VB

Rapid application development is the hallmark of ObjecTrieve/VB. ObjecTrieve/ VB is invaluable for Visual Basic developers as an ISAM-based data manager for building database applications. Add on tools include DbControls and BLOB Manager. From ImageSoft, ObjecTrieve/VB is the data manager for quickly and easily building complete Windows applications.
Price: £320

AM/ST

Coopers and Lybrand's AM/ST was the first application manager developed for Smalltalk/ V and continues to be the premier software engineering environment for developing object-oriented applications for Smalltalk/V

Published by ImageSoft, AM/ST was designed to provide individual developers and development teams with the ability to manage and control large, complex object-oriented applications. Written in Smalltalk/V, AM/ST enhances the Smalltalk system with software engineering and productivity tools.
Price: £385

CommonBase

Combining the power of object-oriented programming with a fully portable class library CommonBase provides the C++ framework for database applications. Developed by ImageSoft CommonBase provides SQL and ISAM with the benefits of C++ and object-oriented programming while providing an alternative to using Embedded SQL or proprietary APIs.

Developing applications for Oracle, Sybase, Gupta, Faircom c-tree, ObjecTrieve, SQL Server, Integra, OS/2 Extended Edition Data Manger, Informix and Novell B-trieve could not be easier. NEW - ODBC support.
Price: DOS and Windows - £765 per database - includes source
OS/2 PM - £765 per database - includes source
UNIX - £2610 per database - includes source

Also available from ImageSoft and Systemstar SoftTools Ltd:
Object/Engineering C++ classes for scientific and engineering. For Windows, Sun Sparc and RS/6000.
cback - C++ optimizer for cfront output.
The Concurrent Language System

MEWEL 4.0

Single Source - Multiple Environments

New release of the **MEWEL** user interface library from Magma Software Systems. A library which allows the developer to write a single set of source code for Windows, DOS Gaphics, DOS Text, OS/2 and UNIX. MEWEL is an implementation of the Windows API.

MEWEL is particularly recommended for zApp 2.0 and other C++ framework users and for expanding class libraries supplied with compilers.
Price: MEWEL/Text for DOS - £365 - with full source £695
MEWEL/GUI for DOS - £365 - with full source £695

Magna*Charter* II

Build flowcharts in minutes!

Magna*Charter* II for Windows lets you build any kind of chart in minutes and includes all standard flowcharting symbols which are displayed as icons or add your own. Use the crowsfeet for database schemas. Features are accessed by drop-down menus, dialog boxes and multiple windows with a wide choice of text styles and sizes. Editing is simple 'cut-and-paste'. A wide range of printers is supported as is PostScript.
Price: £190

STAD5

Other Products Supplied & Supported by Systemstar SoftTools:

zApp	v 2.0 Win,DOS Text or Graphics	£330
zApp	v 2.0 Windows NT	£330
zApp	v 2.0 OS/2	£430
Doc-To-Help		£285
Dolphin C Toolkit		£99
Dolphin Far Memory Manager		£99
Phar Lap 286IDOS-Extender		£365
KnowledgeMan	from: DOS SU	£990

Please check prices at time of order.
Prices do not include VAT or carriage.
All Trademarks Acknowledged.

ISCL

The largest library of C++ Shareware in the World?

Claimed by Image/ISoft to be the largest library of public domain and shareware C++ source code in the world. **ISCL** includes the GNU 386 C++ compiler, a comms library, maths and matrix classes, DPMI programming kit, classes for Btreive and Paradox and a C++ library for building GUIs. **ISCL** contains over 100 MBytes of code and is available only on CD-ROM.
Price: £95

NEW! NEW!

Phar Lap TNTIDOS Extender

(formerly 386IDOS Extender)

Phar Lap's new TNTIDOS Extender lets users of Microsoft Visual C++ 32-bit Edition build 32-bit DOS applications which implement NT features such as DLLs, threads and multi-tasking. Includes Microsoft CodeView 32-bit debugger and Phar Lap 386ISRCBug source level debugger.
Price: £365

NEW! NEW!

InstallSHIELD v2.0

Latest Release of Installation Program for Graphical Applications.

InstallSHIELD has become the standard tool for building bullet proof Windows and OS/2 installation programs. A new 1000 page manual describes the many new advanced features including multi-language support. Existing users may upgrade - call for details.
Price: Windows £295 OS/2 £520

NEW! NEW!

QUICTURE

Compress Graphics in Word for Windows

Quicture is a unique graphics handling utility for Word for Windows. By using Quicture graphics are compressed and off-loaded to disk replacing each one with a placeholder. You can view and print graphics at any time but in draft mode, documents scroll, save and print quickly, taking up less space. From WexTech, the authors of Doc-To-Help.
Price: £75

For more information, demonstration disks or details of our 30 day trial offer call: (0992) 500919

SYSTEMSTAR
SoftTools Limited

1-3 Parliament Square Hertford SG14 1EX
Telephone: (0992) 500919 Facsimile: (0992) 554261

➤ CIRCLE NO. 125

FAST Survey

A survey by the Federation Against Software Theft has revealed that 87% of corporate directors admit that software theft is within their control rather than the publishers. A mere 13% of those surveyed blamed complex licensing arrangements as the reason for unauthorised software copying. FAST, which is made up of US software publishers like Microsoft and Borland, conducted the survey in conjunction with management consultancy KPMG.

New from Select

Select Software Tools has announced that V3.01 of its object-oriented C++ Designer graphical design tool will include a representation of Microsoft Foundation Class V2.0, Microsoft's standard class library for building Windows applications, as well as Borland's OWL Library. C++ Designer V3.01 is now available at the special price of £99.00 for a limited period. Current users of C++ Designer can upgrade for £49.00.

Synon on Windows

Synon, best known as IBM's original AS400 development tools partner, plans to ship a Windows 3.1 generator for its client server series in the first quarter of 1994. The new product is intended to give AS/400 developers the capability to generate client server applications using the AS/400 as a server running OS/400 and the PC as a client running Windows 3.1. The Windows generator will be offered alongside Synon's existing OS/2 version.

Windows File Indexer

Nildram has developed a Windows utility called Key Search which builds an index of all words in a given file, allowing the user to search on keyword. NilDram says it has used the utility itself during its own in-house development to keep track of the many sample files included with Microsoft and Borland compilers.Key Search costs £29.

MIPs

A few months ago, DEC claimed that its Alpha-based PC was the fastest PC on Earth. The latest news from MIPS Technologies puts the MIPS R4400 out in front - 34% faster than the Alpha at running Windows NT applications.

Borland Paradox reborn

IBorland has 'rationalised' its Paradox products. Both Paradox for DOS 4.02 and Paradox for Windows V1.0 have jumped to version 4.5. In addition Borland has split a single product (Paradox for Windows) into 3: Paradox V4.5 for Windows, Paradox V4.5 for Windows Workgroup Edition and Paradox V4.5 Development Edition for Windows.

The basic P4W V4.5 (RRP £99.95) has improved performance over V1.0, support for FoxPro data types, and some ObjectPAL additions, including the ability to export and import data natively. Best of all, it appears to have fixed many of the V1.0 problems.

The Workgroup Edition (RRP £329.95) features OBEX (Object Exchange Technology, allowing sharing of tables, queries and files through mail transports such as MHS, CCMAIL, MCI MAIL), SQL connectivity to Interbase V3.3, Oracle V6.0, Microsoft/Sybase servers, and a workstation configure utility to ease the task of installing P4W on multiple workstations.

The Developer Edition (RRP £459.95, not yet shipping as of mid-October) is likely to contain a Help Compiler, Borland's Resource Workshop, SQL Links Forms Source Code and a Developer Resource Tool kit, but the exact contents could change.

Updates to the DOS 4.5 Product (RRP £499.95) include better mouse support, improved report design and printing, a new debugger, anda customisable speedbar.

A run-time version for P4W was **not** included in these announcements.

Borland seems to be trying to rescue itself from its recent low-price strategy. P4W V1.0 was announced with a high RRP, but its 'limited offer price' of under £100 has been successful at snatching market share. With these announcements, Borland has given up the attempt to jack up the price of the standard product, but should gain much needed revenue from the Developers' and Workgroup products. It is noticeable that the DOS version, where Borland does not face such intense competition, retains its high RRP.

ASK Takes to the Open Road

The ASK Group, parent company of relational database supplier Ingres, last month upped the stakes in the burgeoning database tools market by laying out plans for a major application development framework.

The ASK Group had been widely expected to confirm it was porting its existing ASK/Windows4GL to run against non-Ingres databases, such as Oracle and Sybase, but the scale of the company's plans caught its rivals on the hop.

Codenamed OpenROAD, The ASK Group's framework consists of a new version of Windows4GL, a new code generator, a design methodology and associated services and consultancy offerings.

Steve Weyl, drafted in from Apple Computer earlier this year to head up The ASK Group's tools division, said the ASK mission statement was to become the number one supplier of database independent development tools in the world.

As recently as six months ago, The ASK Group had dismissed the claims of database independent tools suppliers, like Unify and Uniface, and claimed that tools should be tightly integrated to a specific database.

But Pier Carlo Falotti, the ASK Group's chief executive officer, recognised that this view was untenable for a company promoting itself as an open systems supplier and ordered a policy u-turn.

Early next year versions of Windows4GL will be available running against databases from Oracle and Sybase. An Informix version is scheduled for 1995.

The ASK strategy is the latest in a series of tools repositioning announcements by the Big Four UNIX database suppliers. Earlier this year Oracle rebadged its existing tools under the umbrella title of Co-operative Development Environment in a bid to promote a more unified image.

In September Sybase announced plans to develop its own range of tools and a data repository, although these will not be delivered before the end of 1994 at the earliest. Informix plans to ship its long-delayed Informix 4GL++ next year and is also seeking partnerships with high-end Case companies.

C Set++ for the mission critical set.

File Options Actions Configure Tools Windows Switch List Help
Dynamic Call Graph
Erase
Help
Delete
Change
Cancel
Help

Set your eyes on this. C Set++ gives the Workplace Shell quite a workout.

GET SET for incredible 32-bit power. Get set for mission critical reliability. Get set for a range of advanced features. Get C Set++ from IBM.

C Set++ is the most complete object-oriented application development package you can buy for OS/2.

C Set++ lets you create the most advanced, high-performance applications imaginable.

Its 32-bit C/C++ compiler lets you unleash all the power of OS/2, giving you industrial-strength code for your mission critical applications. It has an extraordinary code optimizer, and a full set of class libraries. And there's whole set of other helpful features, like an interactive source level debugger. You also get Workframe/2, a **C Set ++** language-independent tool that lets you customise your own environment.

It's adaptable and flexible – you can use any 16 and 32-bit DOS, Windows and OS/2 tools. With C Set++, it's easier than ever to set your sights on success.

For more information contact your IBM dealer or send us the coupon below.

SIMPLY A BETTER WAY

IBM

For more information on C Set++ send to Carol Samuels, IBM PS Enquiry Centre, FREEPOST, PO Box 32, Alencon House, Alencon Link, Basingstoke, Hants RG21 1EJ.

Name Position Company

Address Postcode Tel

EXEC1

➤ CIRCLE NO. 126

Letters

We welcome short letters on any subject that is relevant to software development. Please write to The Editor, .EXE Magazine, St Giles House, 50 Poland Street, London W1V 4AX. Unless your letter is marked 'Not for Publication', it will be considered for inclusion in this section.

Install wish-list

Sir,

I read with approval Dave Stiles article 'Success with Software'. (*.EXE* September '93). The area of installation is a tricky one: readers of *.EXE* have probably all got their own stories of installations which went wrong.

My personal three 'wish list' items of install programs are:

- First to provide an UNINSTALL (which is very rare - presumably the vendor cannot imagine anyone ever wishing to do without his product!)
- Second, an INSTALL log so I know what files were installed where, and more important, what was changed/added to various config files (eg WIN.INI, OS2.INI or whatever). This is also very useful for network operators wanting to install many copies of a program.
- Third, **helpful** error messages if the installation fails (or if some file is missing or in the wrong place when running the program). A message saying 'File not found' is of little use - what file and where is the program looking?

With these three items in place I feel much happier when installing software since I am then (fairly) confident that I know what's been done to my machine and can undo it later if I decide either to remove the program or to re-install - perhaps on a different drive on my machine.

Roger Orr
London

Windows Paranoia?

Sir,

Whether Peter Dawes of Stockport is suffering from Windows paranoia or not is a matter for his analyst. The reason why Windows is going 'that route' is because it's what the punters want. And punters mean money. It is Mr Dawes prerogative to ignore the hype and spend his money as he pleases. I do wonder, however, when he last looked at a price list if he considers hardware nowadays to carry a 'hefty' price tag.

Mind you, how seriously can you take someone who can bear to be called a cynical Luddite and thinks that a 286 gives blindingly fast performance? I suggest a trip to to his optician with a stop-watch and keep away from Windows.

And who can bear being called a cynical Luddite?

Calum Michael
Strawberry Hill

What? No Stob

Sir,

On receiving the October issue of your magazine, I turned, as I always do, straight to the back page. Shock! Horror! Where's Our Verity? Has she been head-hunted by the former Editor? Has she been offered a 1.5% pay rise and stomped off in disgust? Has someone said something unkind about the size of her bottom?

Please try to have Verity reinstated (not reïnstated, please) as soon as possible. Otherwise I can imagine the emergence of yet another acronym - CURVES (Campaign for the Unconditional Reinstatement of VErity Stob).

Raymond Butler
Croydon
Surrey

Obvious Pointer

Sir,

I believe the program in Figure 1 answers the problem described in Francis Glassborow's article 'Pointers and Arrays in C' (*.EXE October '93*). Although I must confess to being slightly surprised that the answer was not plainly obvious. Therefore it is entirely possible that I have misunderstood the problem altogether...

Jason Hunt
Horsham
West Sussex

```
#include <<stdio.h>>
#include <<stdlib.h>>
char *suit[] =
{
  "Club", "Diamond", "Heart", "Spade"
};
char *fixed[6] [3] =
{
  "AC", "BD", "EF", "GH", "IJ", "YX"
};
void test ( char string[] )
{
  printf("\n%s\n" , string );
}
main()
{
  test (suit [1]);
  test (fixed[0] [1];
  return 1;
}
```

Here is the output for the above program:

```
C:TEST>test
Diamond
▥
C:\TEST>
```

Figure 1 - Solution to last month's ACCU

More UNIX

Sir,

As a fan and programmer of UNIX, I'd rather see more *how to do* articles than *what ZZZZZware's like*, but one has to be pragmatic. There's a place for **both** these types of article. What there should be are less hot-air articles; the space being filled by the more practical aspects of programming. But you can't have a popular journal of algorithms. After all, who would advertise in it? Corporate puff adverts are short on the ground. No magazine is ever completely relevant to *all* its readers *all* of the time.

Programming is a wide and varied skill from kicking the tyres round Prolog to transaction processing with SQL to... well you name it. More often than not you have to come off you're algorithms/methods and get down to some nitty-gritty good old implementation. That's what many programmers spend most of their time on - implementing their ideas.

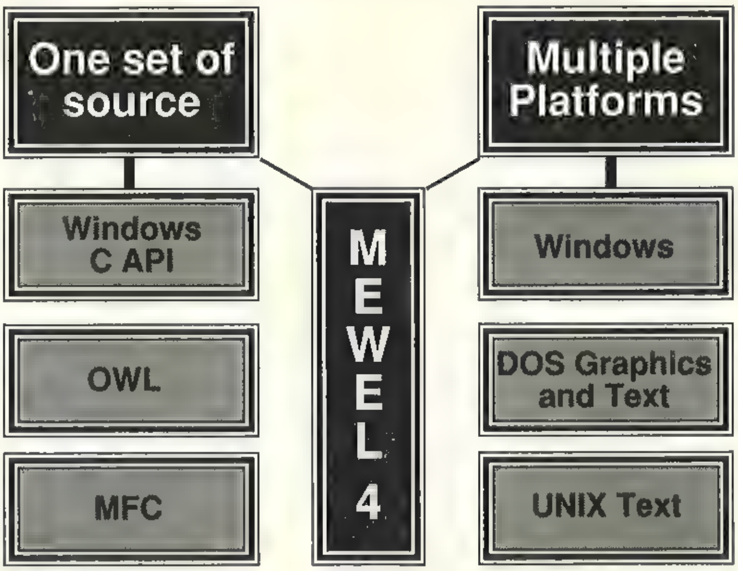
D J Walker-Morgan

Letter of the Month

The writer of the best letter of the month, as judged by the Editor, will receive a .EXE disk of his choice. The best letter is the one printed first. Please note that letters submitted to this page may be edited.

NEW!
4.0 GUI Version

MEWEL Interface Library


One set of source

Multiple Platforms

Windows C API

OWL

MFC

MEWEL 4

Windows

DOS Graphics and Text

UNIX Text

MEWEL API is compatible with the Microsoft Windows API. This means that from a single set of Windows source code, both a Windows and a **MEWEL**-based program can be generated just by recompiling and relinking with the appropriate libraries and header files. With **MEWEL** a Windows program can be extended to any supported environment - currently DOS text and graphics, OS/2 and UNIX.

When using any other cross platform tool you are locked into a proprietary API. **MEWEL** uses the industry standard Windows API.

For C++ programmers, **MEWEL** extends the popular Borland OWL and Microsoft Foundation Classes so that they produce DOS graphics and text applications.

If you have a Windows application and need DOS, **MEWEL** will provide a GUI which is **100% code compatible.**

DOS programmers will find in **MEWEL** all the window objects found in Microsoft Windows - multiple, overlapping, re-sizeable windows, dialog boxes, single and multiple edit fields, listboxes, push buttons, radio buttons, check boxes, scroll bars, combo bars, static text, icons, bitmaps, multiple fonts, multi-level menus, mouse and much more.

MEWEL 4.0 from Magma Systems is distributed and supported in the UK by Systemstar SoftTools. Full source code is available. Call for update information from earlier releases.

MEWEL from Magma Systems - now available from and supported by Systemstar SoftTools Ltd

Call: (0992) 500919 for more information and a demonstration disk.

MAGMA
SOFTWARE
SYSTEMS

SYSTEMSTAR
SoftTools Limited

1-3 Parliament Square Hertford SG14 1EX
Telephone: (0992) 500919 Facsimile: (0992) 554261

MEWAD1

➤ CIRCLE NO. 127

Doc-To-Help®

Doc-To-Help - the hypertext word processor for Microsoft Word for Windows

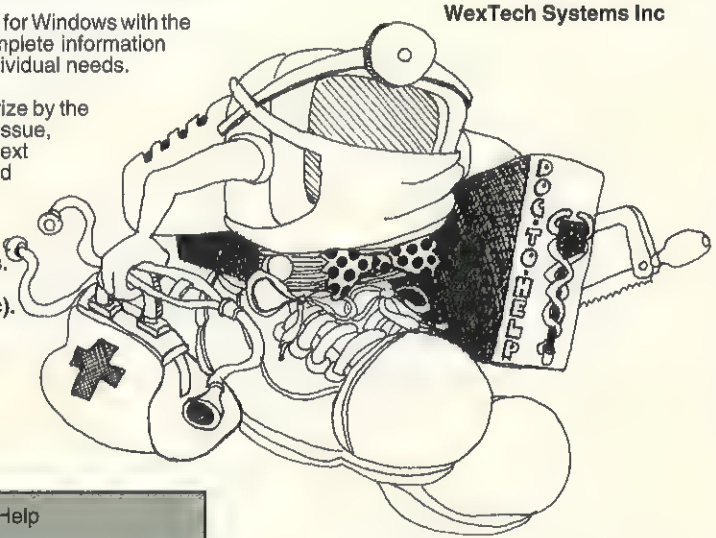
Write commercial quality documentation and convert that documentation into Windows on-line Help, **automatically with *Doc-To-Help*.**

Doc-to-Help is for anyone who distributes information and for anyone who wants that information to have all the impact that can be added by professional design and cutting-edge hypertext presentation.

WexTech Systems Inc

Doc-to-Help synthesizes the desktop publishing capabilities of Word for Windows with the hypertext resources of the Microsoft Help Compiler to create a complete information delivery system that is fast, flexible and completely adaptable to individual needs.

Doc-to-Help is written by WexTech and was awarded the Win100 prize by the US Windows Magazine in the February 1993 edition. In the same issue, *Doc-to-Help* was given the editors choice in a review of leading hypertext tools. The Chairman of the UK Microsoft Users Group has described the product as one of the most exciting that he saw in 1992.



Features of *Doc-To-Help* include:
- **Customizable, professionally-formatted document templates.**
- **Multiple file support.**
- **Automatic formatting options (for screen shots, captions etc).**
- **A powerful indexing utility.**
- **Simplified table creation and formatting.**
- **Easy cross-referencing within the document.**

***Doc-To-Help** is distributed and supported in the UK by Systemstar SoftTools. NEW version 1.5 - £285 + VAT. Call: (0992) 500919 for a demonstration disk.*

NEW- from the authors of Doc-To-Help
QUICTURE
Compress Graphics in Word For Windows
Quicture is a unique graphics handling utility for Word for Windows. By using **Quicture** graphics are compressed and off-loaded to disk replacing each one with a placeholder. Graphics can be viewed and printed at any time but in draft mode, documents scroll, save and print quickly, taking up less space. Available now from Systemstar SoftTools for just £75+ VAT.

SYSTEMSTAR
SoftTools Limited

1-3 Parliament Square Hertford SG14 1EX
Telephone. (0992) 500919 Facsimile: (0992) 554261

DTHAD3

➤ CIRCLE NO. 128

Bugs in the C Standard

No one is perfect. Even the ISO/IEC C standard committee. Derek Jones picks out six of the best bugs.

After what seemed an eternity the ANSI C standard was ratified in December 1989 (it became an ISO standard in 1990 ISO/IEC 9899:1990). To those working on it the contents were as perfect as was humanly possible or at least possible in 50 man years of effort (based on committee time multiplied by number of people attending, plus some time for out of meeting work). Needless to say, requests for interpretation (standards don't *really* contain bugs) were being submitted before the ink was dry on the published document.

Finding the problems

There are several factors working against the detection of problems in a document such as that for C standard. First its sheer complexity: anybody wishing to understand it must undergo a considerable period of study. Prior to its publication, the two groups of people who had studied the language in detail were the standards committee and compiler writers.

It is often true that those closest to something are the least likely to see the flaws in it. So long standing committee members tended to be a very poor source of problem reports.

The best source of interpretation requests tend to be people involved writing compiler-related tools. End users rarely have the time or need to read an entire standard closely. I would go as far as to state that unless you have written a compiler for a language then you do not fully understand it (and yes, I am a compiler writer).

Is the standard no good?

On being told that over 50 interpretation requests have been submitted against the C standard it might be thought that the original authors had done a sloppy job (even more so, perhaps, given that interpretation requests often contain multiple questions).

> ***Unless you have written a compiler for a language then you do not fully understand it (and yes, I am a compiler writer)***

Many of these questions can be broken down into a number of categories:

Questioner does not understand the standard. The answer to these usually involves citing the relevant subclauses.

Questioner disapproves of what the standard states and is asking for a change. The answer to these usually involves saying that the suggested change will be considered next time the standard is revised (this happens about once a decade).

Standard is silent on a particular topic. Strictly speaking, the answer to these questions should be that the behaviour is undefined. Sometimes, 'a detailed reading of the standard will show ...' (committees never admit to making mistakes) that the behaviour is not undefined.

Wording in different parts of the standard appears to conflict. Of course it does not *actually* conflict because a 'detailed reading...' is bound to clear up any ambiguities.

In practice the majority of questions have little impact on programs in the real world. My own view is that C programmers should take comfort from the fact that a group of people is actually trying to create a rigorous language definition. Those interested in rigorous languages might like to know that there have been over 2,000 interpretation requests against the Ada standard (okay, so only one bug was found in the Pascal standard, but 85 have been found in the Extended Pascal standard to date).

The box *A Question of C* presents a selection of bugs in the C standard.

Conclusion

The fact that problems are being found is an indication that the C standard is being used and that people care about it. That most of the problems found in the last three years have been obscure suggests that the average programmer can sleep sound in the knowledge that C does not contain any hidden anomalies (at least from the C point of view Ada programmers may claim that the entire language is an anomaly).

Derek Jones has a background in compiler writing. He has been a conveynor at the C standards panel for the 4 years. He is the Editor of the ISO C addendum and a member of the UK POSIX panel. He spends most of his time checking apps for conformance to various standards.

C++ TRAINING

Simply the best

THE C++ TRAINING PROVIDERS.

Richfords training programme for C++ developers gives you the chance to:

- **Select a truly appropriate course** from our comprehensive training programme.
- **Choose the right entry level**
  We have courses for programmers who already have strong C skills or C++ skills. Also courses for programmers moving to C++ from other backgrounds.
- **Choose which software you learn on**
  Product specific courses are available using software from the market leaders.
- **Benefit from the course leader's commercial experience**
  All our course leaders are also actively involved in the commercial application of the software which they train in.
- **Learn hands-on programming skills in the classroom**
  Richfords courses emphasise the building of practical skills and include many exercises to consolidate the learning process.
- **Use a training organisation which specialises** in software development tools for professional systems developers.
- **Enjoy an intimate learning experience!**
  We keep our class sizes small - no more than six students - so that you can benefit from an appropriate level of supervision during the many practical exercises.

CHECK OUT OUR COURSES...

- **OLE 2 Workshop**
  Using OLE with Visual C++ 1.5 and MFC 2.5
- **Advanced Windows Programming using Visual C++**
  Using 'better C' in DLLs, OLE, ODBC class support with MFC 2.5, writing and using VBX controls.
- **Windows C++ Accelerator**
  From C to WINDOWS/C++ programming in 5 days. Only the very best C programmers need apply.
- **Visual C++ Programming**
  For experienced C programmers interested in C++ extensions and changing their approach to programming in C, using Microsoft Visual C++.
- **Borland C++ Programming**
  For experienced C programmers interested in C++ extensions and changing their approach to programming in C, using Borland C++.
- **C++ for Windows**
  For OWL users, an OWL course. For MFC users, a course designed for that purpose.
- **C++ Foundation**
  New to C++? Starting out with a 'better C'. This is the course for professional programmers starting C/C++ assignments, with no previous C experience.

Courses are run at our Central London Training Centre and on client site.

Call 071 922 8819 for course outlines and bookings.

RICHFORDS

SOUTH BANK TECHNOPARK • 90 LONDON ROAD • LONDON • SE1 6LN • FAX 071 922 8839

➤ CIRCLE NO. 129

A Question of C

The following is a small list of some of the more interesting questions with possible solutions.

Example 1

Here is an example of a reader of the standard (Michael S Ball, doc X3J1190-044) having difficulty in understanding exactly what was intended.

```
typedef int table[];
table one = {1};
table two = {1, 2};
```

The identifier `table` is declared as an incomplete type. Now the initialiser in the declaration of `one` completes the type. Does this mean that subsequent uses of table must contain a single element (thus making the declaration of `two` illegal)?

Answer 1

Subclause 6.1.2.5, page 24, lines 8-9: 'An array of unknown size is an incomplete type. It is completed, for an identifier of that type, by specifying the size in a later declaration ...' In the example the types of the objects `one` and `two` are completed but the type table itself is never completed. Thus the code is conforming.

I know of at least one compiler that used to get this wrong.

Example 2

In this example a reader (Rich Peterson, doc X3J11/90-008) had constructed a piece of code in which the interpretation seemed to be counter intuitive:

```
static int i;     /* declaration 1 */
main()
{
  extern int i;    /* declaration 2 */
  {
    extern int i;  /* declaration 3 */
  }
}
```

Is this a conforming program? If not why not?

Answer 2

This example contains undefined behaviour. Where? I hear you ask. Well subclause 6.1.2.2 states: 'If the declaration of an identifier for an object or a function contains the storage class specifier `extern`, the identifier has the same linkage as any visible declaration of the identifier with file scope.'

The first declaration of `i` has file scope. This file scope declaration is visible to the second declaration. But the second declaration 'hides' the first declaration from the third declaration. Thus the third declaration declares identifier `i` to have external linkage, because no file scope declaration is visible.

Later on in subclause 6.1.2.2 we have: 'If, within a translation unit, the same identifier appears with both internal and external linkage, the behaviour is undefined.'

This probably counts as an interesting curiosity: unless you happen to be trying to compile some automatically generated code in which the generator got overly carried away with declaring identifiers.

Example 3

This example (Derek Jones, doc X3J11/90-056) is a case of ambiguous wording in the standard which offers two possible interpretations. What is the status of the following code fragment?

```
#define f(a, b) a + b
#if f(1,
        2)
#endif
```

The C standard quotes the following:

Subclause 5.1.1.2, page 5, line 37: 'Preprocessing directives are executed and macro invocations are expanded.'

Subclause 6.8, page 86, lines 2-5 say: 'A preprocessing directive ... and is ended by the next newline character.'

Subclause 6.8.3, page 89, lines 38-39 say: 'Within the sequence of preprocessing tokens ... newline is considered a normal white-space character.'

In the above example is the newline after the `f(1,` considered to end the `#if` directive or is it treated as a whitespace character (thus allowing `2)` to be appended?

Answer 3

The committee decided that the wording on page 86 stated the intent and that this was not altered by the wording on page 89.

The net result of this ruling is that line splicing ('\') has to be used if a `#if` directive takes up more than one line.

Example 4

The following example (same ref as the previous example, but actually discovered by Bruce Blodgett) is probably one of the largest problems so far found in the standard.

```
int x(T (U));
int y(T (U (int a, char b)));
```

In the first declaration `U` is the type of a parameter to a function returning type `T`. But what about the second declaration? From subclause 6.5.4.3, page 68, line 2 we have: 'In a parameter declaration, a single `typedef` name in parenthesis is taken to be an abstract declarator that specifies a function with a single parameter, not as redundant parentheses around a declarator.'

So in the second declaration U could be either a redundantly parenthesised name of a function that takes a parameter-type-list and returns type T or the type returned by a function which takes a parameter-type-list, that, in turn, is the single parameter of a function returning type T.

The committee answered this question by stating a general principle that had been intended, but not stated in the standards document. Ideas?

Answer 4

The response to this question introduced a general principle, not found in the standards document: 'Whenever a `typedef` name could be taken as such in a declaration, it is so taken.'

This question is probably of more significance to compiler writers than end-users. However, users are becoming more sophisticated in their use of functions.

Example 5

Here is something (same ref as the third example above) for those of you who think you understand how the preprocessor works:

```
#define f(a) a*g
#define g(a) f(a)

main()
{
  int i = f(2)(9);
}
```

The standard supports two possible macro expansions for the preprocessing-token sequence, `f(2)(9)`. What are they and what was the committees response to the request for a definitive answer?

Answer 5

The two possible sequences are `2*f(9)` and `2*9*g`. The committee then went on to say: 'The behaviour in this case could have been specified, but the committee has decided more than once not to do so. [They] do not wish to promote this sort of macro replacement usage.'

This is nasty. The committee is trying to prevent the use of some C constructs by not defining them. The net result is that some poor user is likely to be very surprised on finding two compilers giving different answers and probably very annoyed when told that the standard allows such behaviour.

Example 6

This is my favourite (not just because I asked it).

```
#define hash_hash # ## #
#define mkstr(a) # a
#define in_between(a) mkstr(a)
#define join(c, d) in_between( c hash_hash d)
char * j = join(x, y);
```

Does `'xy'` or `'x ## y'` get assigned to `j`?

At first glance this code is not conforming. However, a closer reading of the standard shows that it is perfectly legal. All of the above definitions are required. How was this code created in the first place? It originally arose from a discussion I had with the editor of the standard who told me that it was not possible for # to occur legally outside of strings and pre-processing directives. I did not agree. In fact, this is about the only case where they can occur, so I guess that he was almost right.

Answer 6

The string `'x ## y'` is assigned to `j`. The rationale given was that expanding `hash_hash` produces a new token consisting of two sharp signs, but this new token is not the cocatenation operator.

Okay. I admit it. This example is ultra obscure. I would claim that it is the hardest example (brain power needed to understand) of preprocessor goings on that can be written.

Example 7

Here is an example (same ref as third example above) for those of you involved in processing input.

```
int i; double x;
i=fscanf(stdin, "%5le", &x);
```

Given the input characters `1.2e-x` what characters will be processed?

The relevant citations are:

subclause 7.9.6.2, page 137, lines 15-16: 'If conversion terminates on a conflicting input character, the offending input character is left unread in the input stream.'

subclause 7.9.6.2, page 135, lines 31-33: 'An input item is defined as the longest matching sequence of input characters, unless that exceeds a specified field width, in which case it is the initial subsequence of that length in the sequence.'

subclause 7.9.6.2, page 135, lines 38-40: 'If the input stream is not a matching sequence, the execution of the directive fails; this condition is a matching failure.'

So there are two possibilities. Either the characters `1.2` or the characters `1.2e-` will be read, but which?

Answer 7

With the given input the characters `1.2e+` will be consumed and the value `0` will be assigned to `i`.

In case you did not already know, doing anything other than reading complete lines and then using purpose written routines to parse the input is bad news in C.

The virtues of assembler

Using the Huffman decoder algorithm as an example, Chris Hall compares the advantages of assembler over C.

In this article I hope to illustrate that sometimes it is better to code an application in assembler rather than assume, without considering any implications, that C is the language of choice. The example I will present is an implementation of Huffman decoding written in both C and assembler.

Huffman

Huffman codes are used to transmit symbols. The actual definition of a symbol is dependent on the application. For instance, when transmitting a text file, the symbols might be the individual characters, or pairs of characters, or even words.

Each symbol is represented by a codeword where the number of bits in the codeword for a given symbol is inversely proportional to the probability of that symbol appearing. Symbols that appear often have shorter codewords, so the total number of bits transmitted is reduced.

A Huffman code is an optimal, or minimum redundancy, code. Decoding is straightforward, and does represent a significant overhead compared to reading a simple byte stream.

Example Huffman Code

Suppose we have eight symbols' space and the letters 'a' to 'f'. Now in a file containing 7,450 of our symbols, if we represent each symbol as a three bit value, the file would be 22,250 bits long.

To construct a Huffman code for this file we take the number of times each different symbol occurs in the file, and use the algorithm to allocate a codeword length to each. (How that is done is outside the scope of this article.) Having allocated codeword lengths, we then allocate codeword values, as shown in Figure 1. Using this code the file would be 16,450 bits long.

A little C

To decode a Huffman code the reader must identify each codeword, and translate it into the required symbol. Identifying a codeword involves reading the number of bits required by the shortest codeword. Depending on the value of these bits, the codeword is either complete, or requires more bits. If more bits are required, then the number of extra bits required for the next longer codeword are read, and appended to the bits already read. The codeword is now either complete, or requires further bits.

Figure 2 lists a C function to read a codeword. It uses a table with one entry per codeword length used, where each entry gives the codeword length, and the last codeword value at this length. The entries are arranged in ascending codeword length order. For our example the table would contain:

Length	Last Codeword
1	0_2
3	101_2
4	1110_2
5	11111_2

For each codeword read we must find the symbol value. The codewords at each length forms a sequence. So from

Symbol	Occurs	Bits	Codeword
space	4,000	1	0_2
e	1,000	3	100_2
a	800	3	101_2
f	500	4	1100_2
c	400	4	1101_2
b	300	4	1110_{b2}
d	250	5	11110_2
g	200	5	11111_2

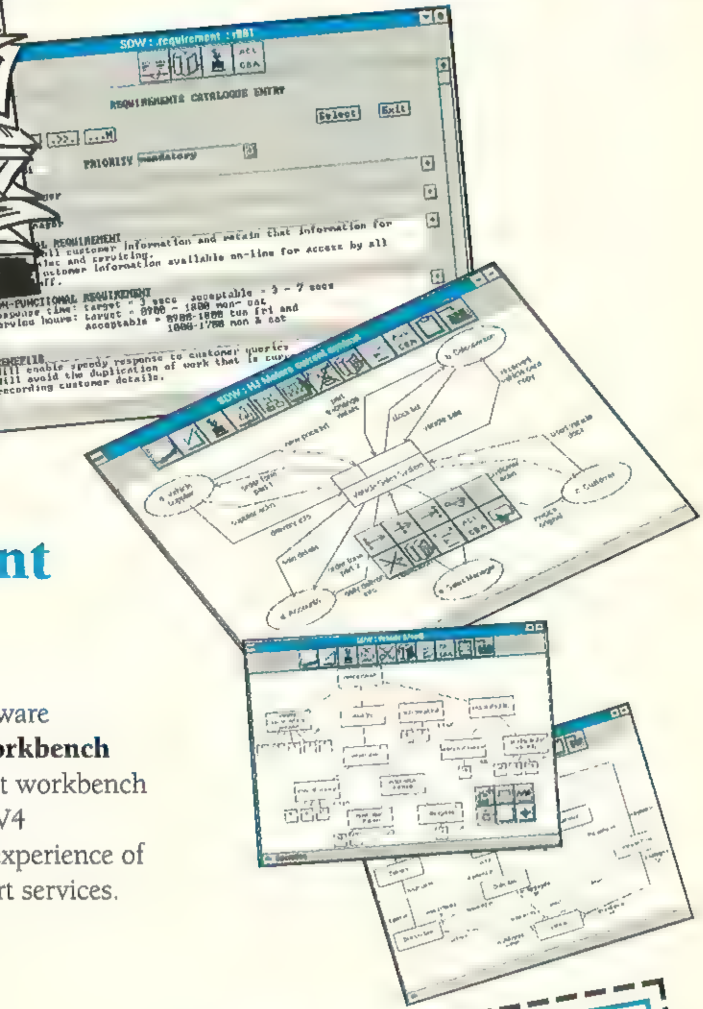
Figure 1 - Example Huffman encoding

```
static struct
{
  int             code_word_length ;
  unsigned long  last_code_word ;
} huff_decode[32] ;

// read and return huffamn codeword
unsigned long huff_read ()
{
  int i, prev_length, this_length ;
  unsigned long code_word ;

  prev length = 0 ;
  code_word = 0 ;
  i = - 1 ;
```

```
  do {
    this_length =
      huff_decode[++i].code_word_length;
    code_word =
      (code_word <<
         (this_length - prev_length))
         + read_bits(this_length -
                     prev_length) ;
    prev length = this length ;
  } while (code word >
      (huff_decode[i].last code word));

  return(code_word) ;
} ;
```

Figure 2 - Outline of Codeword Reader

Taking the pain out of using SSADM



```
SDW : requirement : r001
REQUIREMENTS CATALOGUE ENTRY
PRIORITY  mandatory
L REQUIREMENT
ill customer information and retain that information for
ales and servicing.
ustomer information available on-line for access by all
aff.
NON-FUNCTIONAL REQUIREMENT
Response time: target = 3 secs  acceptable = 3 - 7 secs
Service hours: target = 0700 - 1800 mon- sat
               acceptable = 0700-1800 tue fri and
                            1000-1700 mon & sat
BENEFITS
Will enable speedy response to customer queries
Will avoid the duplication of work that is cur
recording customer details.
```

SDW for SSADM from Hoskyns - the systems development experts

Now you can achieve major improvements in your software development projects - with **System Development Workbench (SDW) for SSADM**. This powerful, easy to use analyst workbench has been specifically designed to help you use SSADM V4 effectively. It's backed by Hoskyns' extensive practical experience of developing IT solutions and a complete range of support services.

At only £1250 (Windows or DOS) SDW for SSADM is unbeatable value. It gives you:

- heavyweight support for SSADM V4
- a central data repository for true team working
- an intuitive mouse driven interface
- user controlled consistency and validity checking

Implementing and using SSADM need not be a headache. Hoskyns can help by providing not just the tools but also SSADM experts to support, train and advise. So if you're looking for a quicker, more effective way to use SSADM, return the coupon, or call us on 071-434 8111 today.

HOSKYNS
CAP GEMINI SOGETI

□ I would like more information on SDW for SSADM.

□ Please contact me to discuss my SSADM requirements.

Name ____________________

Job Title ____________________

Organisation ____________________

Address ____________________

Post Code __________ Tel. No __________

Return to: Project Systems Division FREEPOST
SW5779. Hoskyns Group plc, Hoskyns South Bank,
95 Wandsworth Road, London SW8 2HG

➤ CIRCLE NO. 130

the n_i codewords at the i^{th} length we can extract an ordinal in $0..n_{i-1}$. This ordinal can be used to look up the corresponding symbol directly.

Now, we can simplify things a little. First, instead of holding the codeword length in our table, we store the extra length ie the number of further bits to read, dispensing with all the `prev_length` nonsense!

Next, the first codeword at a given length is the prefix for codewords at that length, implying that the prefix for a given length is a combination of the prefixes for all previous lengths. Thus we can strip the prefix progressively (instead of reading the entire codeword and stripping the prefix at the end). Additionally, we can combine this with the detection of completed codewords. To see how it works we must look at the way prefixes combine. The prefix at the first length (p_1) is zero. The prefix at the i^{th} length (p_i) is then:

$$p_i = (p_{i-1} + n_{i-1}) << (b_i - b_{i-1})$$

where n_i is the number of symbols at the i^{th} length, and b_i is that length.

During decoding, when we have read the b_i bits required for the i^{th} codeword length, we have the value so far, v_i. We examine v_i to see if we have a complete codeword, or whether more bits are required.

Suppose that we can arrange for v_i to be the b_i bits read, minus the prefix p_i. If v_i is a codeword, then v_i will be in $0..n_{i-1}$. Furthermore, if v_i is not a codeword, then we can subtract n_i before we append the bits to construct v_{i+1}, making v_{i+1} the b_{i+1} bits read, minus the prefix p_{i+1}.

The first prefix is zero, so v_1 is trivially the b_1 bits, minus the prefix p_1. We can, therefore, arrange for all subsequent v_i to be the b_i bits read, less the prefix p_i.

So, we replace the last codeword entries in the decoding table by the number of codewords at each length. We use this number both to detect whether a complete codeword has been read, and progressively to strip the prefix.

$0..n_1$-1	symbols at first length
$n_1..n_1+n_2$-1	symbols at second length
$n_1+n_2..n_1+n_2+n_3$-1	symbols at third length
and so on.	

Figure 3 - Indexing symbol translation table

```
int const max_huff_symbols = 256;
int const max_huff_length  =  32;

typedef unsigned char HUFF_SYMBOL;

typedef  struct HUFF_CODEWORD_ENTRY
{
  int extra_length;
  int count;
} HUFF_CODEWORD ENTRY;

typedef  struct HUFF DECODE_TABLE
{
  HUFF_CODEWORD_ENTRY
    codewords[max_huff_length];
  HUFF_SYMBOL
    tx_table[max_huff_symbols];
} HUFF_DECODE_TABLE;

typedef struct HUFF_READ_FILE
{
  HUFF_DECODE_TABLE * bits_decode;

  unsigned char bits_byte;
  unsigned char bits_count;

  unsigned char * buff_pointer;
  unsigned char * buff limit;

  unsigned char * buff address;
  unsigned int    buff size;

  int file_handle;
} HUFF_READ_FILE;


HUFF_SYMBOL huff_read_symbol
(
  HUFF_READ_FILE *from
)

{
// Read next Huffman coded symbol
// from the given file.

  int ordinal_origin, current_value, bits;
  HUFF_CODEWORD_ENTRY *ptr;

  ordinal_origin = 0;
  current_value  = 0;

  for (ptr = from->bits_decode->codewords;
       current_value >= 0; ptr++)
  {
    bits = ptr->extra_length;

    while (bits > from->bits_count)
    {
      current value =
        (current_value << from->bits count)
         + ((from->bits_byte) >>
         (8 - from->bits_count));
      bits -= from->bits_count;

      if (from->buff_pointer ==
          from->buff_limit)
      {
        int  amount_read;

        amount_read =
          read(from->file_handle,
               from->buff_address,
               from->buff_size);
        if (amount_read < 1) exit(255);

        from->buff_pointer =
          from->buff_address;
        from->buff_limit   =
          from->buff_address + amount_read;
      }

      from->bits_byte  =
        *(from->buff_pointer++);
      from->bits_count = 8;
    }
    current_value =
      (current_value << bits)
      + (from->bits_byte >> (8 - bits));
    from->bits count -= bits;
    from->bits_byte <<= bits;

    ordinal_origin += ptr->count;
    current value  -= ptr->count;
  }

  return
  (
    from->bits_decode->
    tx_table[current_value + ordi-
nal_origin]
  );
}
```

Figure 4 - Huffman Code Reader, in C

Note that by removing the prefix at each stage we can guarantee that the value so far, v_i, never exceeds the total number of symbols minus one. So if there are 256 symbols in the alphabet, then the accumulated value never exceeds 255.

The final simplification: for the translation to the symbol value we arrange to have a single direct look-up table, which is indexed as illustrated in Figure 3.

If we add the code necessary to read bits from an input file, the complete C code is now as shown in Figure 4. This is for symbols which can be expressed as bytes. The declarations of `SYMBOL` and `max_symbols` can be changed, provided that `max_symbols` does not exceed the maximum value for an `int`.

Max Codeword Length

The value of 32 for `max_lengths` requires some justification. For a given symbol the codeword will be approximately $-\log_2(p)$ bits, where 'p' is the probability of the symbol appearing. We can only say 'approximately' because the codeword must be a whole number of bits. We can say definitely that no symbol will have a codeword longer than `CEILING(-log2(p))+1` bits.

So, a maximum codeword length of 32 bits will cope with a minimum probability of 2^{-31}. If no more than 2^{31} symbols are sampled when establishing the probabilities, the minimum probability (other than zero) will be no less than 2^{-31}. This is not, usually, difficult to arrange.

```
PBYTE  TYPEDEF PTR BYTE
;
;      Structure for Huffman Reader "Stream Descriptor" ;
HUFF_READ_FILE  STRUCT  1
  DECODE        PBYTE  ?  ; Pointer to the Decode Table
  BITS_BYTE     BYTE   ?  ; Current byte
  BITS_COUNT    BYTE   ?  ; Number of bits in current byte
  BUFF_POINTER  PBYTE  ?  ; Pointer to next byte to read
  BUFF_LIMIT    PBYTE  ?  ; Pointer to last byte to read + 1
  BUFF_ADDRESS  PBYTE  ?  ; Address of the Buffer
  BUFF_SIZE     WORD   ?  ; Size of Buffer in bytes
  FILE_HANDLE   WORD   ?  ; File Handle for reading file
HUFF_READ_FILE  ENDS
;
    PUBLIC  ?huff_read_symbol@ZAEPAUHUFF_READ_FILE@@Z
              ; "MS decorated" name for
              ;  huff_read_symbol function ;
;==================================================
?huff_read_symbol@ZAEPAUHUFF_READ_FILE@@Z PROC FAR
;================================================== ;
;      Read next symbol from given Huffman Input Stream ;
    MOV   BX, SP
    MOV   BX, [BX+4]        ; Get BX = Address of
                            ; Huffman Stream Descriptor ;
    PUSH  SI                ; Must be preserved for MS C++ ;
;      Point at the Decode Table & pick up
;      current byte & bits ;
;        BX = Address of Huffman Stream Descriptor ;
    MOV   SI, (HUFF_READ_FILE PTR [BX]).DECODE
                        ; SI := Address of the Decode Table ;
    MOV   CX, WORD PTR ((HUFF_READ_FILE PTR[BX]).BITS_BYTE)
                        ; CH := BITS_COUNT ; CL = BITS_BYTE ;
;      The decode loop
;
;        SI = Current address in Decode Table
;        DX = Number of symbols at previous length (DH = 0) ;

;        CH = BITS_COUNT
;        BX = Address of Huffman Stream Descriptor ;
;        AH = Value so far
;        AL = BITS_BYTE
;
    XOR   DX, DX            ; No symbols at (no)
                            ; previous length
    XOR   AX, AX            ; Value so far !
    MOV   AL, CL            ; AL := BITS_BYTE
;
$10huff_read_symbol:
    ADD   SI, DX            ; Step to next Codeword length ;
    MOV   CL, [SI]          ; CL := Bits wanted
    INC   SI
;
    SUB   CH, CL            ; Assume we have all the
                            ; bits we need
    JB    $30huff_read_symbol  ; J if we were wrong ;
$20huff_read_symbol:
    SHL   AX, CL            ; AH := New value so far ;
    MOV   DL, [SI]          ; DX := Number of symbols at this
length
    INC   SI
;
    SUB   AH, DL            ; Subtract symbols count
                            ; from value so far
    JNB   $10huff_read_symbol  ; J if not yet finished ;
;      We have completed the codeword
;
;        SI = Address of TX VECTOR
;        DX = Number of symbols (DH = 0)
;        CH = BITS_COUNT
;        BX = Address of Huffman Stream Descriptor ;
;        AH = Ordinal - Number of Symbols
;        AL = BITS_BYTE
;
    MOV   CL, AL            ; CL := BITS_BYTE
    MOV   WORD PTR ((HUFF_READ_FILE PTR[BX]).BITS_BYTE), CX
                            ; Put back: CH = BITS_COUNT ;
                            ; CL = BITS_BYTE ;
    ADD   DL, AH            ; DX := Ordinal
    ADD   SI, DX            ; Point at symbol value ;
    MOV   AL, [SI]          ; Fetch
;
;      Finished !
;
;        AL = Symbol
;
    POP   SI
    RET
;
;      We fo not have enough bits to hand.
;      Use what we have, then fetch more. ;
;        SI = Current address in decode table
;        DH = 0
;        CH = BITS_COUNT - Bits wanted
;           = - Bits we will need after taking what we have ;
;        CL = Bits wanted
;        BX = Address of Huffman Stream Descriptor ;
;        AH = Value so far
;        AL = BITS_BYTE
;
$30huff_read_symbol:
    ADD   CL, CH             ; CL := BITS_COUNT
    SHL   AX, CL             ; AH := New value so far ;
;      Now fetch the next byte from the input file ;
;        SI = Current address in decode table
;        DH = 0
;        CH = - Bits we need now
;        BX = Address of Huffman Stream Descriptor ;
;        AH = Value so far
;
    PUSH  SI
;
    MOV   SI, (HUFF_READ_FILE PTR [BX]).BUFF_POINTER
                             ; Get Buffer Pointer
    CMP   SI, (HUFF_READ_FILE PTR [BX]).BUFF_LIMIT
                             ; Address of last byte + 1
    JE    $50huff_read_symbol  ; J if Buffer is empty ;
$40huff_read_symbol:
    LODSB                    ; AL := next byte
;
    MOV   (HUFF_READ_FILE PTR [BX]).BUFF_POINTER, SI
                             ; Put back Buffer Pointer ;
    POP   SI
;
    XOR   CL, CL
    SUB   CL, CH             ; CL := Bits we need now ;
    ADD   CH, 8              ; CH := 8 + (- Bits we need now)
                             ;     = Bits left after what we
                             ;       need now
    JMP   SHORT $20huff_read_symbol
;
;      Need to refill the buffer
;
;        DH = 0
;        CH = - Bits we need now
;        BX = Address of Huffman Stream Descriptor ;
;        AH = Value so far
;
$50huff_read_symbol:
    MOV   CL, AH             ; Salt away Value so far
    PUSH  CX
    PUSH  BX
;
    MOV   DX, (HUFF_READ_FILE PTR [BX]).BUFF_ADDRESS
    MOV   CX, (HUFF_READ_FILE PTR [BX]).BUFF_SIZE
    MOV   BX, (HUFF_READ_FILE PTR [BX]).FILE_HANDLE
    MOV   AH, 03FH
    INT   021H               ; Ask DOS for more
;
    POP   BX
    POP   CX
;
    JC    $60huff_read_symbol  ; J if failed
;
    OR    AX, AX
    JZ    $60huff_read_symbol  ; J if hit EOF
;
    MOV   SI, DX             ; SI := Address of first byte ;
    ADD   AX, DX
    MOV   (HUFF_READ_FILE PTR [BX]).BUFF_LIMIT, AX
                             ; Set new buffer limit ;
    MOV   AH, CL             ; Restore AH = Value so far
    XOR   DX, DX             ; Restore DH = 0
;
    JMP   SHORT $40huff_read_symbol
;
;      Failure reported by DOS or hit EOF
;
$60huff_read_symbol:
    MOV   AX, 04C01h
    INT   021h               ; Give Up !!
;
?huff_read_symbol@ZAEPAUHUFF_READ_FILE@@Z  ENDP
```

Figure 5 - Huffman Code Reader, in assembler

THE WORLD'S ONLY OBJECT-ORIENTED SYSTEM SOFTWARE IS NOW AVAILABLE IN THIS BIG BOX.

Byte Magazine said it's "...probably the most respected piece of software on the planet." *Business Week* said that it's "...years ahead of its potential rivals, such as Microsoft Corp.'s Cairo ..." NEXTSTEP™ for Intel® 486 and Pentium® Processors is the world's first object-oriented system software. Now it's here. Polished and perfected. Shrink-wrapped and shipping. In user and developer versions. For desktop and portable systems.

THE OBJECT IS **THE ADVANTAGE**™.

➤ CIRCLE NO. 132

Now you've seen our big box, simply fill out the little box below for further details. Or, ring us on 081-565 0005 or fax on 081-565 0016

NeXT Computer UK Limited, Somerville House, 50a Bath Road, Hounslow, Middlesex TW3 3EE

Name: .... Company: ....

Address: .... Postcode ....

Telephone: .... Fax: ....

.EXE 11/93

© 1993 NeXT Computer, Inc All rights reserved. NeXT, NEXTSTEP, the NeXT logo and The Object Is The Advantage are trademarks of NeXT Computer, Inc. All other trademarks mentioned belong to their respective owners.
Business Week, *January 25, 1993 reprinted with permission.* Byte Magazine, *October 1992 reprinted with permission*

Faster and Smaller

The Huffman reader is a modestly long function. Using the Microsoft Visual C++ compiler (V1.00), with all possible optimisations turned on (favouring speed over size) the function above compiled to 208 bytes for a 386 (using the Medium Model, 16-bit code).

I recorded `huff_read_symbol` in assembler, as shown in Figure 5. (The code as shown assembles using MASM V6.) As with C++ there is no error recovery. To avoid using definitions files the DOS calls are coded using 'magic numbers'. The assembler version is 118 bytes, compared to 208 for C++, so occupies 45% less space. A significant reduction!

To compare these versions of the Huffman decoder I wrote two small programs. The first reads a small file into memory, and then decoded the memory, copying into a buffer one hundred times, without writing away the result, so eliminating any disk I/O overhead. The second program decodes a large file, reading the encoded file from disk and writing the decoded result back to disk. The programs were run on three machines, under DOS V5.0, and the elapsed times per byte decoded were recorded as shown in Figure 6.

On all the machines tested, using the assembler version of the decode function is twice as fast as the C++. Surprisingly, the 486 fared little better than the 286, despite it being apparently optimised for compiled code.

Why assembler is better

1 **An intelligent use of registers.**

The loop in the Huffman decoder has everything it needs in registers. This is a significant factor even on the 486, despite its ability to access a variable in (cached) memory much faster than earlier x86 processors.

2 **Cunning use of shifts.**

The core of the codeword reader is a shift of bits from the input into the current value. In the assembler version this is done in one operation. The compiler generates three separate shifts.

3 **The use of the carry flag.**

The assembler version uses the carry produced by the subtraction from the byte wide `current_value` to signal the end of codeword reading. In the C++ an `int` must be used for `current_value` in order to get the loop termination.

4 **The removal of the `ordinal_origin`.**

This saves some work and allows the loop to have all variables in registers.

5 **The use of jumps out of the main path for the exception cases.**

Generally, each time round the loop only a few bits will be read (typically one or two). So the current input byte is likely to satisfy the read. The assembler code jumps out of the loop if another byte must be read. The compiler generates a jump which will be taken if another byte need not be read. (On the x86 processors a jump take is more expensive than a jump not taken: on earlier processors, this is much more expensive.)

For the record

One way in which the assembler improves on the C++ version is by small changes to the decoding data structure. First, the codeword information and symbol values are interleaved, removing the need to maintain the 'ordinal origin' value, by making the one register point to the codeword information and the translation information. Second the codeword extra length and codeword count have been reduced from words to bytes. This works fine for the byte symbols that the program manipulates. For symbols larger than a byte we would need a rather different piece of assembler.

There is one wrinkle however. If all 256 symbols have the same length (8 bits) then we have a problem. We cannot represent a symbol count of 256 in one byte. This case could be detected elsewhere, and the data read directly, avoiding the redundant Huffman reader altogether. Alternatively, the code will work quite happily if the decode table has a first entry of 255 codewords of 8 bits and a second of one codeword requiring no extra bits.

I did try a C++ version using the same data structure, but it came out about 10% slower! This is one of the pitfalls of optimising C code: you cannot necessarily tell what the compiler will do with the 'optimisation'.

Memory-to-Memory decoding, 1,594,700 bytes decoded (from 941,200 bytes)

	486SX-25	*386DX-33*	*286-10*
C++ version:	12.2	18.0	99.4
ASM version:	6.23	9.41	46.4
ASM/C++	0.51	0.52	0.47

Disk-to-Disk decoding, 821,206 bytes decoded (from 535,105 bytes)

	486SX-251	*486SX-252*	*386DX-33*	*286-10*
C++ version:	14.1	16.0	19.1	120
ASM version:	6.89	9.97	9.71	66.9
ASM/C++	0.49	0.62	0.51	0.56

Where the 486SX-251 had SMARTDRV running, while 486SX-252 did not.

Figure 6 - Time (in micro-secs) to decode a byte

Observations

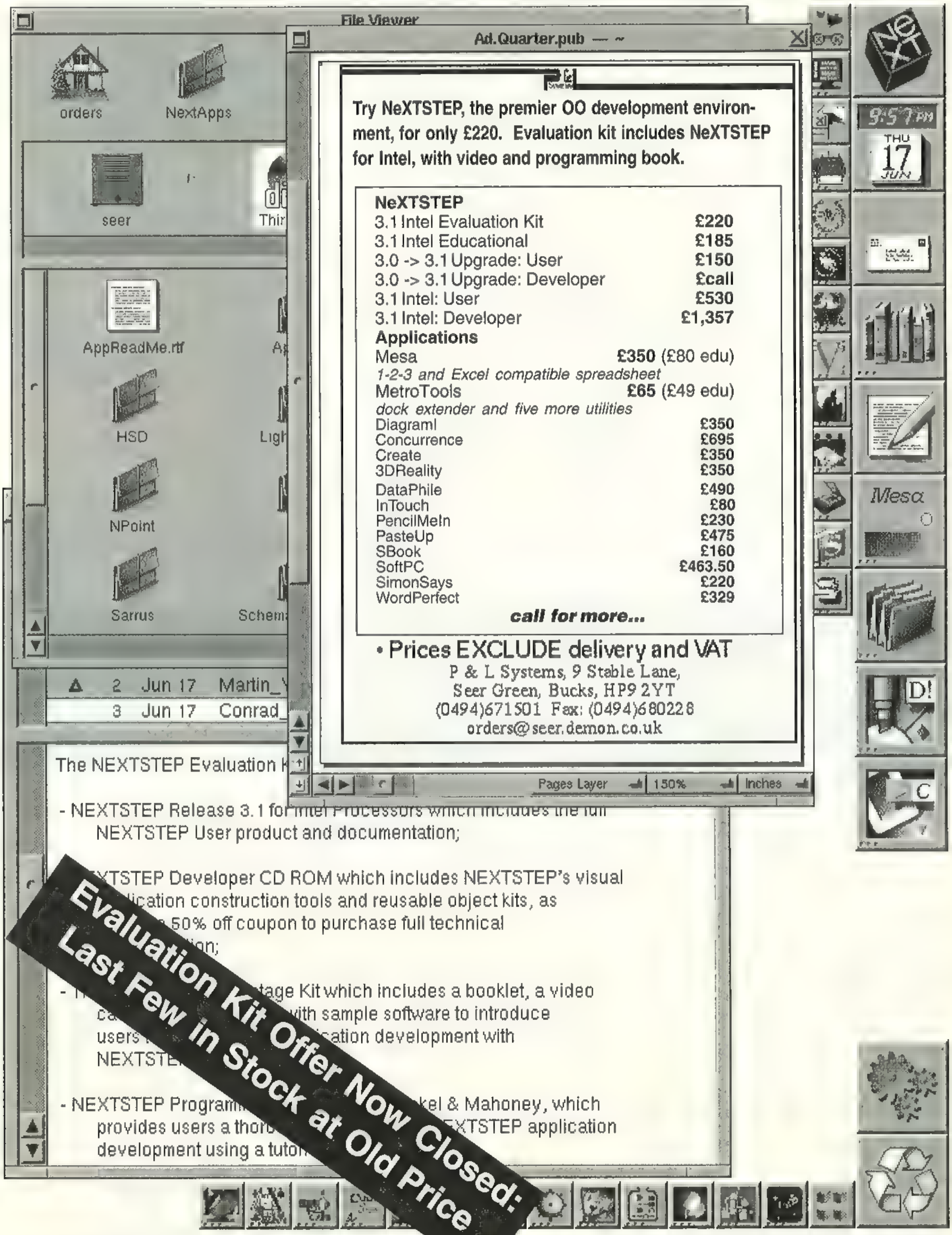
Coding this single function in assembler gives a significant performance improvement, and a marked reduction in size. Why has the human code generator outperformed the compiler?

Some of these are specific to the application, but are generally indicative of the superiority of the human code generator. In fact, over larger sections of code it is possible to do even better by:

- Long term use of registers - Compilers tend only to keep data in registers locally. In assembler one can arrange to carry key pieces of data in registers for extended periods (ie the pointer to a data structure). This saves time moving the data to and from memory, and the space required for the movement instructions.

P & L SYSTEMS

Software, Training and Services for NeXTSTEP



P&L Systems

Try NeXTSTEP, the premier OO development environment, for only £220. Evaluation kit includes NeXTSTEP for Intel, with video and programming book.

NeXTSTEP	
3.1 Intel Evaluation Kit	**£220**
3.1 Intel Educational	**£185**
3.0 -> 3.1 Upgrade: User	**£150**
3.0 -> 3.1 Upgrade: Developer	**£call**
3.1 Intel: User	**£530**
3.1 Intel: Developer	**£1,357**
Applications	
Mesa	**£350** (£80 edu)
1-2-3 and Excel compatible spreadsheet	
MetroTools	**£65** (£49 edu)
dock extender and five more utilities	
Diagraml	£350
Concurrence	£695
Create	£350
3DReality	£350
DataPhile	£490
InTouch	£80
PencilMeIn	£230
PasteUp	£475
SBook	£160
SoftPC	£463.50
SimonSays	£220
WordPerfect	£329

call for more...

- Prices EXCLUDE delivery and VAT

P & L Systems, 9 Stable Lane,
Seer Green, Bucks, HP9 2YT
(0494)671501 Fax: (0494)680228
orders@seer.demon.co.uk

△ 2 Jun 17 Martin_
3 Jun 17 Conrad_

The NEXTSTEP Evaluation K

- NEXTSTEP Release 3.1 for Intel Processors which includes the full NEXTSTEP User product and documentation;

XTSTEP Developer CD ROM which includes NEXTSTEP's visual lication construction tools and reusable object kits, as 50% off coupon to purchase full technical ion;

- T tage Kit which includes a booklet, a video ca with sample software to introduce users ication development with NEXTSTE

- NEXTSTEP Program kel & Mahoney, which provides users a thoro XTSTEP application development using a tutor

Evaluation Kit Offer Now Closed: Last Few in Stock at Old Price

- Prices EXCLUDE delivery and VAT
- Cash or cheque with order

P & L Systems, 9 Stable Lane, Seer Green, Bucks, HP9 2YT
(0494)671501 Fax: (0494)680228 orders@seer.demon.co.uk

(0494)671501

➤ CIRCLE NO. 133

- Passing parameters in registers - This is where the big savings come. Passing parameters on the stack is a wonderfully general mechanism. It does, however, result in a lot of pushing things onto the stack and sorting out stack frame pointers. Well constructed code, with lots of medium size and small procedures/functions can suffer badly from this effect. The size of small functions, and the cost of calling them, can become dominated by the cost of passing a few parameters.

In assembler one arranges for parameters (or at least the important ones) to be passed in registers, if at all possible, ensuring that the caller will naturally have the data in the required register.

- Keeping local variables in registers.

Often local variables are limited in scope even within a procedure/function, so it is straightforward to keep most local variables in registers. (A little pushing and popping of registers may be required to keep some variables on the stack temporarily.)

- Small procedures/functions.

One of the advantages of high-level languages is that a single line of code may generate a lot of machine instructions. This is also one of the disadvantages. An operation which appears trivial in a high-level language may be repeated rather than factored out into a function. The assembler programmer can see the cost, and may save a lot of code by writing a small procedure or function.

Other Experience

We recently used a bought-in component, written in C. In order to integrate this with our word processor we took the assembler output of the C compiler, and replaced all system and many library calls.

We spent one day rapidly working through the code, changing some obvious parameter passing from stack to register: particularly where a parameter was pushed from a register by the caller, and promptly loaded into a register by the callee. This promptly removed 3 KB of code from a 30 KB module.

Conclusion

I chose the decoding of Huffman codes for this study because it is prototypical of functions for which assembler is most appropriate. First, the decoder is likely to be the innermost part of some larger function, so qualifies as speed critical code. Second, it is largely nitty-gritty machine level operations.

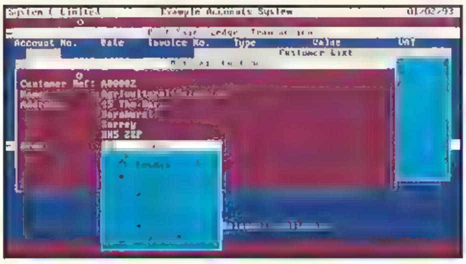
However, the result, code which is half the size and twice the speed, is not unique. Indeed, because this is such a small piece of code it does not show all the advantages of the intelligent use of registers.

So, if you are faced with a project that doesn't meet its space or speed requirements, then assembler is what you need.

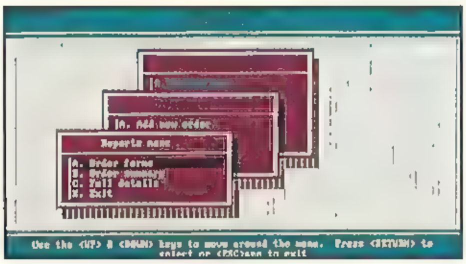
.EXE

Chris Hall is the Managing Director of Locomotive Software. He can be contacted by telephone: 0306 742140, or by Email: chris@locomotive.com.

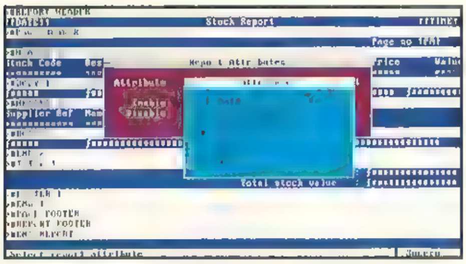
DOUBLE *your* MONEY



Sycero supports panels, movable windows, VGA colours, scrolling screens, tick boxes, radio buttons and much more


A. Add new order
Reports menu
A. Order forms
B. Order summary
C. Full details
X. Exit
Use the <UP> & <DOWN> keys to move around the menu. Press <RETURN> to select or <ESC>ape to exit

Use the definable program templates to produce different program types including menus, file maintenance, reports and enquiry programs



The report generator automates layout design, nested loops, print attributes, ten levels of sub-total, range checking, wild card matching and up to 99 formats per report

Developing Clipper applications with the all-new Sycero is at least twice as fast as coding manually. This means that Sycero developers can write applications at half the cost or less.

And Sycero's new program templates, together with the ability to add ordinary Clipper code within the development environment, gives you the same control over your programs as you have when coding in native Clipper. You gain the speed without losing any power.

Add this to Sycero's GUI-like user interface, automated documentation, ability to handle the most complex of programs and its automatic networking and you end up with an excellent Clipper development environment.

ALL THE POWER OF CA-CLIPPER IN HALF THE TIME

Sycero 3·0, an all-new release, is even faster and easier to use. Its data dictionary organises your data files and indexes. The new screen painter lets you design a modern and elegant interface that end users will love. No database application is complete without reports, and Sycero's powerful combination of automatic features makes even the most complex ones simple to construct.

Of course seeing is believing, so come and see for yourself at one of our regular free seminars. In just a couple of hours we will show you how you can double your productivity.

> *"It goes without saying that Sycero lets you develop many times faster. More important to us are the dramatic improvements in the user interface of our applications, the automated networking and the ability to modify systems easily once they had been installed"*
>
> **Fred Rees**
> systems accountant, Vodafone

System

Send to:

System C Limited
60-61 High Street
Maidstone
Kent
ME14 1SR

Tel: 0622 691616
Fax: 0622 691241

Please send me details of Sycero 3·0 and your next free seminar

NAME ____________________

COMPANY ____________________

ADDRESS ____________________

____

____

POSTCODE ____________________

TELEPHONE ____________________

FAX ____________________

I am/am not a CA-Clipper user

SYCERO

Cheshire Cats and Abstract bases

Charles Weir of Object Designers discusses two ways of removing compilation dependencies between C++ classes.

C++ has a major weakness when it comes to compilation. At present there are no code management tools or development environments capable of distinguishing between the public and private sections of class declarations. So if you change the private data or private definitions within any of your class declarations, the compiler will need to recompile every file that uses that class - even if the public interface has not changed at all. A small change to one of the private function declarations in a base class could well require the complete recompilation of a vast system - even though nothing significant has changed.

There are two different approaches to solving this problem. They are known as *abstract base classes,* and *Cheshire Cat classes.* This article looks at the implications of both approaches, and examines each in detail. As an illustration, we shall attempt to implement a `Motor` object, which represents a motor controlled by our software. The external interface can start and stop this motor, and set its speed. A motor is represented by an instance of the C++ class `Motor`.

Abstract Bases

An abstract base class declares only pure virtual functions which contain declarations but no implementation.

An abstract base class declares only pure virtual functions which contain declarations but no implementation.

Using this approach, the public header file for an object declares its supported functions using an abstract base class. In a different header or implementation file, a derived class defines the actual implementation. Thus the abstract base class creates an *abstract interface* to the class. Code that uses an instance of the class can refer to it solely in terms of the abstract base class and need not have any compilation dependencies on the implementation class. An example is given in Figure 1.

Base file:

```
class Motor
{
public:
  // Start the motor
  virtual void start() = 0;
  // Stop the motor
  virtual void stop() = 0;
  // Set the speed
  virtual void setSpeed( float ) = 0;
};
```

Implementation file:

```
class MotorImplementation : public Motor
{
public:
  // Stores name for debugging
  MotorImplementation( char* name );
  void start();
  void stop();
private:
  // ... Implementation specific data
  //     goes here.
};
```

Figure 1 - Motor class header and definition

One problem with it is that there is no way to create an instance of the class without including the definition of the implementation class; the compiler cannot create a call to `new` unless it 'knows' the size of the object it is constructing. One way around this is to define one or more creation functions as part of the abstract interface. These are static functions which create a new instance of the implementation and return a pointer of the appropriate type. Since they are deputising for the constructors, they will take the same parameters as their corresponding constructor functions (see Figure 2).

```
class Motor
{
public:
  // Answers a new motor instance.
  static Motor* create(char*);
  // Starts the motor.
  virtual void start() = 0;
  // Stops the motor.
  virtual void stop() = 0;
  // Sets the speed.
  virtual void setSpeed( float ) = 0;
};
```

Figure 2 - Implementation independent Creation functions

It's tempting to make the create function inline. We cannot do that, though, without giving knowledge of `MotorImplementation` to clients, thus defeating the entire purpose of the

System Science

• London's specialist supplier of Software Development Tools for DOS, Windows and Unix •

UNIX

SCO Unix Op. Sys SCO Development Sys.
SCO Open Desktop - NEW 3.0 SCO Xsight
SCO TCP/IP & NFS FTP PC/TCP
Interactive Unix Op System/ Architect Series
Informix 4GL & SQL RM Cobol-85
LPI Fortran LPI C++ SCO, SUN, HPUnix, etc
C++ Views - new version with GUI Designer
Word Perfect for Unix PC-NFS
November Special - C++/Views & FREE C++
... many more for 386 Unix and Sun, HP, DEC Risc etc

Mathematics

Derive - new School licenses available
Derive XM - NEW for 386 extended memory
Mathematica 2.2 Win, MAC, Workstation, Student Ver
MathCad Windows Lindo Linear Prog
Origin Graphing with 3D & Contour for Windows
Mathtype Win PC Matlab for Win

Windows Development

(see Microsoft , Borland, Symantec, Watcom)

Data Entry Workshop by Turbopower	£169.00
Win /Sys Lib	£99.00
HALO Imaging Lib Windows	£395.00
Smalltalk/V Windows	£285.00
PCX ProgToolkit Windows	£169.00
CASE:W & KASE:W VIP	£call
WindowsMaker	£545.00
INSTALLShield ver 2.0	£265.00
Winstall	£245.00
Optlink Windows NEW 5.0 with comp	£265.00
Plug-In Edit Control	£375.00
WinGen - screen generator	£125.00
Doc-To-Help for Help file prep	£call
Control Palette Windows by Blaise	£110.00

VB Value

3D Graphics Tools	£125.00
Spread/VB	£195.00
Q&E Multilink/VB	£325.00
CodeBasic 5.0 with CodeReporter	£139.00
QuickPak Prof for Windows	£149.00
3D Gizmos from MicroHelp	£79.00
Muscle/Win from Microhelp	£155.00
VB/Tools from MicroHelp	£139.00

CrossDevelopment

2500AD Cross Assemblers	£175.00
Hitech C Cross Z80, 6805, 68HC11..	£495.00
Hi-Tech C 68K inc Asm, Debugger	£625.00
Introl C or Mod-2, Aztec C68K	£call

For Fortran

Lahey F77L-EM/32 new faster 5.1	£765.00
Graphoria for F77L-EM/32	£245.00
Salford FTN 77 Developer	£745.00
PC/Interacter (scrns & graphics)	£325.00
Halo Professional for Fortran	£245.00
Ingraf - with source	£245.00

- Please call if the item you are looking for is not listed.
- Call for our comprehensive catalogue
- Prices are exclusive of VAT.
- Prices (except upgrades & specials) include delivery to mailand UK. Sameday London delivery at cost.
- Prices are subject to change - please call to check.
- VISA, ACCESS & Mastercard credit cards welcome with telephone orders.

E&OE

Microsoft

Visual C/C++ - 32 bit Ed CD*NEW*	£299.00
Visual C/C++ PE - disk or CD-ROM	£call
Delta - Windows Version Control now shipping	
MultiMedia Viewer Pub Kit	£245.00
MS Test for Windows - *New ver 2.0*	£245.00
MS Windows DDK 3.1	£275.00
MS Macro Assembler 6.1	£99.00
Visual Basic - Windows 3.0 SE	£99.00
Visual Basic Windows 3.0 PE	£229.00
MS Fortran 5.1	£115.00
Fortran PowerStation - 32 bit	£235.00
Visual Basic for MS-DOS Prof	£225.00
MS Project for Windows	£385.00

upgrade to Visual C/C++ & new Visual Basic 3.0

Authorised Languages Dealer

Borland

Borland C++ & AFW 3.1	£345.00
Borland C++ for OS/2 2.0 NEW	£call
Turbo C++ Visual Edition	£75.00
Turbo C++ DOS & C++ Video	£85.00
Turbo Pascal 7	£85.00
Borland Pascal with Objects	£225.00
Paradox Engine - *ver 3.0*	£119.00
Brief Programmers Editor	£119.00

Authorised Languages Dealer

Nu-Mega

Bounds Checker 2.0 for DOS	£159.00
BoundsCHECKER/W *NEW 2.0*	£195.00
Soft-ICE/W (Windows)	£295.00
Soft-ICE (for 386 DOS)	£295.00

Special pricing on Nu-Mega combi packs

Intersolv *(Authorised)*

PVCS Version Manager*new 5.1*	£345.00
new WINDOWS OS/2 version, call for upgrades	
multi-user for DOS Networks, OS/2 , Sun, HP9000, RS6000	
PVCS Config Builder better make than Make	£175.00
PVCS Reporter for Win or OS/2	£125.00

MKS

MKS Toolkit -Unix tools DOS or OS/2	£195.00
MKS RCS - NEW Windows Ver	£call
MKS Lex & Yacc *for C, C++ & TP*	£195.00

Lifeboat

Dan Bricklin's Demo II - prototyping, demos etc	£175.00
Advantage Disassembler	£195.00
Essential COMMs & Graphics tkts	£call

Tools and Editors, Etc

CodeWright Editor for Win new ver	£185.00
Ed for Windows Editor- new 2.0	£145.00
Kedit (Xedit for PC) ver 5	£110.00
Easy Case Professional -DOS or WIN	£call
EasyCase System Designer DOS	£945.00
MetaDesign - Flowcharts, DFDs etc	£295.00
PC Logo	£60.00
RTLink Plus	£345.00

WATCOM

C/C++ ^{32} - new for DOS32, Win,OS/2 , ADS etc	£375.00
C /C++- ^{16}- new DOS & WIN	£325.00
Fortran 77 ^{32} - NEW for dos32,Win, ADS etc	£395.00
SQL Dev Edition - Windows or DOS	£275.00
VX*REXX visual REXX for OS/2	£call

C/C++

Symantec C++ DOS , Win, 32 - *NEW*	£call
Zortech C++ OS/2 2.0 - *NEW 3.1*	£175.00
Metaware High C/C++ 386	£call
Symantec C++ for MAC	£345.00

C++ Libs

Rogue Wave C++Libs	£call
C++/Views - Win, OS/2, Motif	£call
Zinc Interface -DOS, Win or OS/2	£call
Poet (OOPS Database)	£call
Comm++ by Greenleaf - NEW Ver	£189.00

C Datafile & Comms

CodeBase 5.0 /CodeBase++	£195.00
****** SPECIAL PRICE UNTIL 30 /11/93 *******	
Ctree Plus from Faircom	£call
Essential Comms	£210.00
Greenleaf CommLib - 4.0 for DOS/Win	£235.00
C Asynch Manager (Blaise)	£135.00

C Screens & Graphics

CScape with Look & Feel - NEW 4.01	£345.00
Panel Plus II (now withWin support)	£275.00
Greenleaf DataWindows - DOS	£235.00
PCX Programmers Tool	£169.00
GX Graphics	£169.00
GIF Programmer's Toolkit	£169.00
GX Printer - NEW	£225.00
HaloProfessional	£245.00
MetaWindows - NEW	£call

Misc. Libs

Install Professional	£245.00
C Tools Plus	£110.00
Bar Code Library- source	£375.00

... many more tools

Pharlap

TNT DOS-Extender 32 bit SDK	£365.00
286 DOS-Extender SDK	£365.00
386 ASM/Link/LOC	£975.00

Databases

Access by Microsoft	£295.00
Fox Pro - DOS or Windows	£call
Paradox - DOS or Windows	£call
Crystal Reports for Windows	£145.00

What's New!

BoundsCHECKER/W ver 2.0	£call
and with MS TEST for Windows	£special
Symantec C/C++ DOS & Win - special offers	
Visual C/C++ - 32 bit Win32 & Win/NT	£299.00
upgrade from MS VC/C++Prof until 31 Oct 1993	
Watcom VX*REXX for OS/2	£call
EasyCASE System Designer WIN	£1045.00
Windows/NT now shipping	

3-5 Cynthia St
London N1-9JF
Fax: 071 837 6411

071 833 1022

• Software • Specialists • Software • Specialist • Software • Specialists •

➤ CIRCLE NO. 136

```
Class Header:
class MotorImplementation;     // Forward declaration.

class Motor {
public:
  Motor( char* );
  ~Motor();
  void start();
  void stop();
  void setSpeed( float );
protected:
  MotorImplementation * p;
};
```

Implementation file defining the class `MotorImplementation`; each function in `Motor` calls the corresponding function in `MotorInterface`.

```
MotorImplementation {
friend class Motor;
protected:
  MotorImplementation( char* );
  void start();
  void stop();
  void setSpeed( float );
private:
  // Private functions and data go here...
};
Motor::Motor( char* s )
  : p( new MotorImplementation( s ) ) {}
Motor::~Motor()     { delete p; }
void Motor::start() { p->start(); }
void Motor::stop()  { p->stop(); }
void Motor::setSpeed( float s ) { p->setSpeed( s ); }
```

Figure 3 - Motor class with cheshire Cat interface

construction. So the create function goes in the implementation file:

```
Motor* Motor::create
(
  char* name
)
{
  return new
    MotorImplementation(name);
}
```

The syntax for creating a new `Motor` on the heap is now rather non-standard, but certainly keeps the users of the item from needing any knowledge of the `MotorImplementation` class. It would be used as follows:

```
Motor* pMotor1 =
  Motor::create("Motor 1");
```

Although there is no way to create a `Motor` instance on the stack, in static memory or embedded within another object; the `Motor` class behaves identically to one implemented without an abstract interface. There is a cost, however, since all its functions must be virtual, but in practice there are few circumstances where this cost becomes significant.

Cheshire Cats

There is another way to achieve a similar separation between interface and implementation. You may remember that in the book *Alice in Wonderland*, the Cheshire Cat fades away until just the Smile remains. With the Cheshire Cat approach, the Smile is the interface, visible much further than the remainder of the implementation. The name, was contrived by Glockenspiel, which uses it within the company's own class libraries.

The class declaration in the public header file looks normal, but its only item of private data is a pointer to the instance of the implementation class. Each public function in the interface class merely passes the request on the corresponding function in the implementation class; hence these classes are often called 'handle' classes, since they provide a handle to the underlying implementation. An example is given in Figure 3.

Only instances of the `Motor` class, (and of subclasses of the `MotorImplementation` class) will access the functions in `MotorImplementation`, so these functions are protected. This approach has avoided the irritations of the abstract interface approach: we can create a Cheshire Cat `Motor` on the heap, in static memory, or on the stack. In use, it behaves identically to the simple implementation. The cost is higher, though. First, the mechanism to redirect calls to the implementation is more costly both in code written and in execution time. Each function requires two C++ function calls. A virtual function is equivalent to about one-and-a-half non-virtual calls. Second, the Cheshire Cat class must have a destructor even if the implementation has none. Finally, there is an additional heap allocation and deallocation for each instance created.

Inheritance

Suppose we want to have another version of `Motor` called `IndustrialMotor` with additional functionality. This has a different (although compatible) implementation of the `start` function, and is intelligent enough to know its actual speed. It has an extra function, `getSpeed`, for this

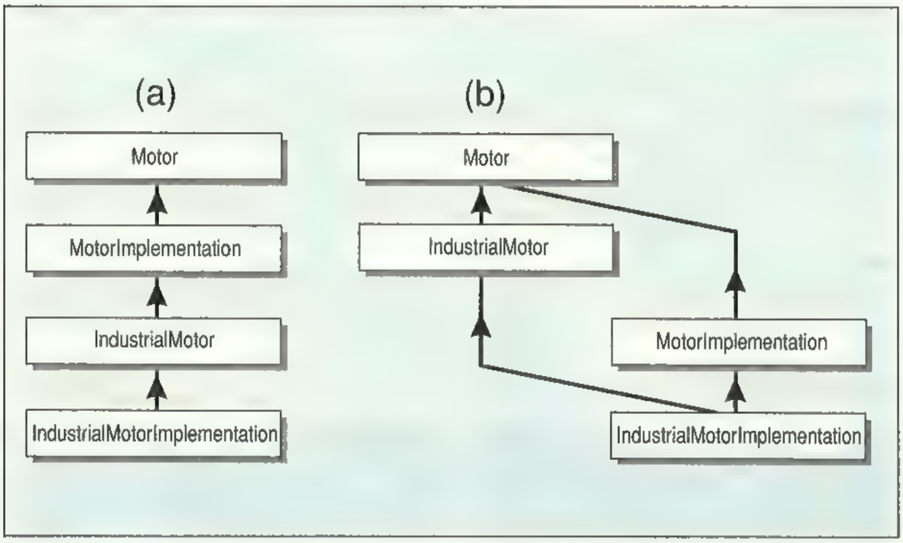
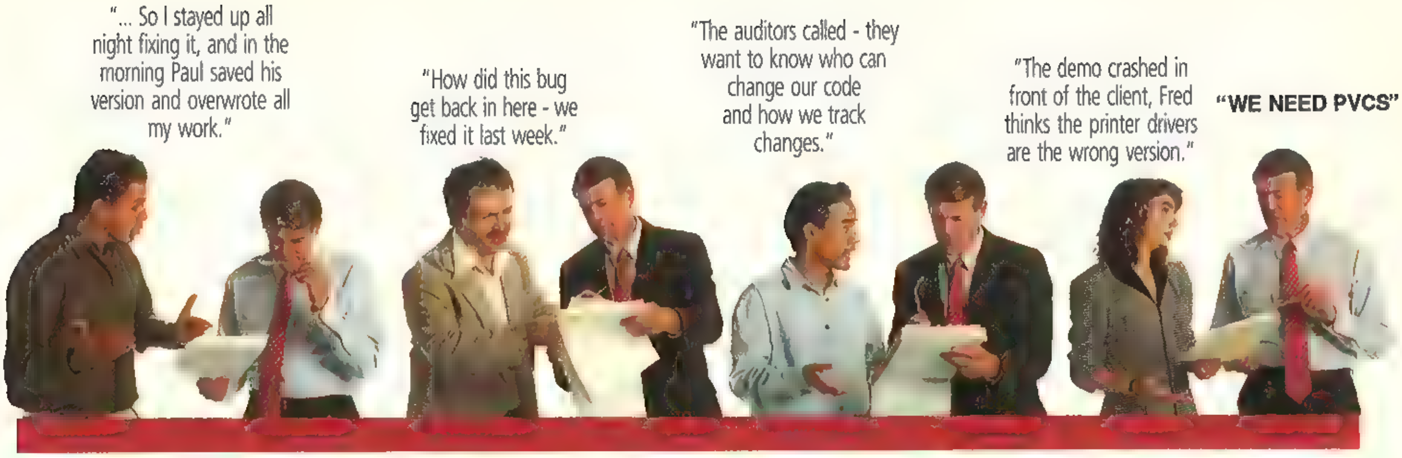


Figure 4 - Two ways to inherit from an abstract interface


"... So I stayed up all night fixing it, and in the morning Paul saved his version and overwrote all my work."
"How did this bug get back in here - we fixed it last week."
"The auditors called - they want to know who can change our code and how we track changes."
"The demo crashed in front of the client, Fred thinks the printer drivers are the wrong version."
"WE NEED PVCS"

Protect Your Code, Your Projects, and Your Reputation

Our customers know PVCS saves them money, time, and anguish. The painful scenarios PVCS avoids are familiar to all developers. Because *all developers do configuration management.* If you manage more than a few modules of code, have more than one person working on a project, or do LAN-based development, then you already do it. *The only question is how well.* And the answer affects the reliability of your projects, the productivity of your developers, and your reputation with your users.

The PVCS Series offers a complete solution to configuration management, that's important because development is getting more complex – not less. Implementing a partial solution today is simply creating tomorrow's problem.

PVCS Version Manager oversees changes to the elements that comprise a software system, tracking who made changes, when and why they made them, and permitting any previous revision of a module or any prior version of the entire software system to be reconstructed at any time. PVCS Version Manager also includes facilities to control parallel development and to reconcile any conflicting changes made to a module. PVCS enables a productive workgroup approach to development, because it is specifically architected for the LAN, and because the feature set and GUI interfaces specifically support distributed client server development. Modules are retrieved for work from wherever they are stored – developers don't have to remember where the archives are located. This simplifies and encourages code reuse, even across operating system boundaries.

PVCS Configuration Builder automates execution of software development tools, permitting systems to be built with absolute repeatability. This increases productivity, but more importantly, entirely eliminates a common source of errors. PVCS Configuration Builder can now embed "footprints" of vital historic information into compiled objects and can act conditionally on footprint information.

PVCS Production Gateway synchronizes the mainframe library to the LAN version archives, providing the control and auditability of the mainframe without bottlenecking the productivity of the LAN environment.

PVCS Developer's Toolkit delivers full PVCS functionality to your development environment or to your applications in the form of DLLs, import libraries and header files. The scope and quality of the PVCS DTK meets IBM and other programming standards for API implementation.

PVCS Reporter lets you quickly produce reports using an advanced graphical interface (in MS-DOS or Presentation Manager) to access the wealth of project information managed by PVCS. You can use predefined queries or build your own. PVCS Reporter queries can span multiple projects in distributed architectures to select, merge, sort, display and print exactly what you need.

PVCS is the undisputed leader in Configuration Management for LANs. It operates seamlessly across MS-DOS, OS/2, VMS, and a host of Unix environments.

The PVCS Series is so accommodating that it snaps right into existing projects. You don't have to wait until you start a new project to implement it. More than a few of our customers have purchased PVCS to salvage projects that were dissolving into chaos... and they've been successful. Cure your biggest headaches before they happen.

Call now for a FREE copy of "Cost Justifying Software Configuration Management".

Phone or fax for details of the PVCS Services. Tel 071 625 5255 Fax 071 624 9404

Evaluation Copies Available

New Windows / OS/2 GUI Version 5.1

Readmar Systems
LIMITED
239 Kilburn Park Road London NW6 5LG

➤ CIRCLE NO. 137

purpose. So we create a derived class `IndustrialMotor` and redefine functions as necessary. Clearly we want to keep the same scheme (abstract interface, or Cheshire cat) for the derived class as for the base class.

Abstract Inheritance

From the illustration in Figure 4 we can see that the abstract interface approach offers two possibilities for the inheritance hierarchy

Scheme (a) is straightforward, using only single inheritance. However, it does not achieve the desired result. To use `IndustrialMotor`, a file must include the definition of `MotorImplementation`, which is exactly what we are trying to avoid. So scheme (a) fails.

Scheme (b) uses multiple inheritance. In fact, it needs more than just simple multiple inheritance. Normally, each instance of a derived class contains an instance of the base class. Thus with simple multiple inheritance, an instance of `IndustrialMotorImplementation` will contain two instances of the class `Motor`, one from `MotorImplementation`, and one from `IndustrialMotor`. The solution to this problem is 'virtual inheritance' as shown in Figure 5, which tells the compiler to create only one instance of the shared base class.

Although it will work in this example, further derived classes will encounter the same problem. A more useful definition of `IndustrialMotorImplementation` is provided in Figure 6.

```
class MotorImplementation :
  public virtual Motor
{ /* ... */ };

class IndustrialMotor :
  public virtual Motor
{ /* ... */ };
class IndustrialMotorImplementation :
  public IndustrialMotor,
  public MotorImplementation
{ /* ...  *  };
```

Figure 5 - Virtual Inheritence from an abstract interface

```
class IndustrialMotorImplementation :
  public virtual IndustrialMotor,
  public virtual MotorImplementation
{ /* ...  */ };
```

Figure 6 - Another stab at abstract inheritence

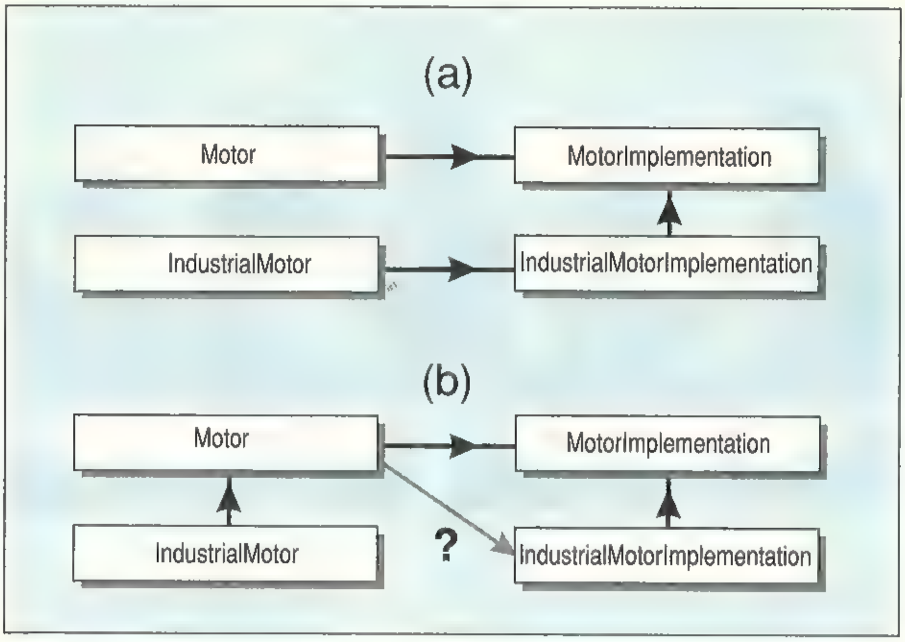


Figure 7 - Two ways to inherit a Cheshire Cat interface

Unfortunately, there are some restrictions on what we can do with virtual inheritance. The compiler will call only the default constructor on the base class. Constructor initialisation lists cannot pass parameters to the constructors of virtual base classes. So if we need to pass parameters to the constructor of the base class, we're stuck. In addition, this approach will stress the implementations of our compiler and debugger, since multiple virtual inheritance is a wonderful arena for obscure compiler bugs. Although the best current implementations are unlikely to give us serious problems, we must be prepared for surprises. So we can achieve inheritance, but at the cost of added complexity, compiler stress and additional restrictions on what we can do with inheritance. How does the Cheshire Cat approach fare by comparison?

In the book Alice in Wonderland, the Cheshire Cat fades away until just the Smile remains

Cheshire Cat Inheritance

The situation is rather different with the Cheshire Cat approach. If we try to get inheritance we end up with several possibilities. There are a number of questions which must be asked. Should the handle classes inherit from each other? And what are the implications of this inheritance? Figure 7 illustrates two types of inheritance for Cheshire Cat classes.

In (a) the `Handle` classes do not inherit, making the implementation of the classes simple, at the cost of duplicating all of the public functions in `Motor` in the derived class. However, there is a problem if we need a generic pointer that can point either to a `Motor` or an `IndustrialMotor`. The abstract interface approach provided this polymorphism. But here it is not possible at all. So approach (a) will not help us if we want the inheritance to be visible to users of the objects. The approach shown in (b) gets around this problem. However it immediately raises problems of its own. We want a new instance of `IndustrialMotorImplementation` when we create a new `IndustrialMotor`, and a new `MotorImplementation` when we create a new `Motor`. But, because the `IndustrialMotor` constructor calls the `Motor` constructor, we will end up with a `MotorImplementation` in both cases; this is not the behaviour we intended.

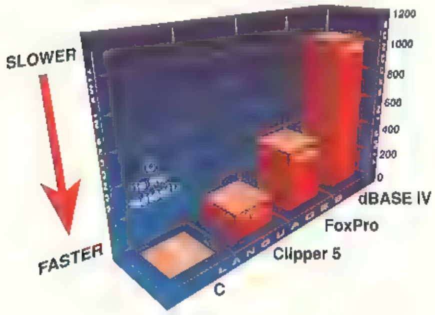
Discover why FoxPro, Clipper, and dBASE were all written in C.

There is a good reason why your database language was developed in C. In fact, there are many good reasons.

C code is small. C code is fast. C code is portable. C code is flexible. C is the language of choice for today's professional developer. With the growing complexity of database applications, C is a realistic alternative. Now with CodeBase 5.0, you can have all the functionality, simplicity and power of traditional database languages together with the benefits of C/C++.

C speed - fast code, true executables...

FoxPro, Clipper, and dBASE were written in C primarily for speed. But those compilers don't really compile, they combine imbedded language interpreters into your .EXE. Now that's slow. For dazzling performance you need the true executables of C. With CodeBase you get the real thing, C code. Consider the following statistics, from the publisher of Clipper:



"Sieve of Erastothenes"
Benchmark for Prime Number Generation
Shows C to be incredibly faster !

C size - small executables, no added overhead...

FoxPro, Clipper and dBASE would like you to believe you need their entire development system to build database applications. But remember, those products are all written in C. So why do you need to lug all their extra code around? You don't. CodeBase is a complete DBMS, in C. No fat executables stuffed with unused code. No runtime modules. No royalties. Just quality C code. CodeBase is just what you need.

C portability - ANSI C/C++ on every hardware platform...

No other language exists on more platforms than C/C++. Why rewrite your entire application for DOS, Windows, Windows NT, OS/2 or UNIX? With CodeBase the complete C source code is included, so you can port to any platform with an ANSI C or C++ compiler. Now and in the future.

dBASE Compatible data, index and memo files...

You want the industry standard. You need compatibility. Sure, dBASE is the standard, but every dBASE compatible DBMS product uses its own unique index and memo file formats. Only CodeBase has them all: FoxPro (.cdx), Clipper (.ntx), dBASE IV (.mdx) and dBASE III (.ndx). Now it's your choice, we're compatible with you.

Announcing CodeBase 5.0

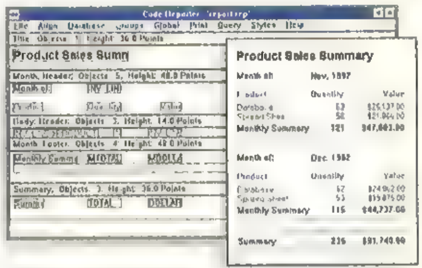
The power of a complete DBMS, the benefits of C

NEW - Multi-user sharing with FoxPro, Clipper and dBASE...

Now your multi-user C/C++ programs can share data, index and memo files at the same time as concurrently running FoxPro, Clipper and dBASE programs. No incompatibilities. No waiting.

NEW - Queries & Relations 1000 times faster...

CodeBase 5.0 now lets you query related data files with any logical dBASE expression. Our new Bit Optimization Technology (similar to FoxPro's Rushmore technology) uses index files to return a query on a 1/2 million record data file in just a second. Automatically take advantage of this query performance by using our new CodeReporter:



Title Objects 1 Height 36.0 Points
Product Sales Summ
Month Header; Objects 5, Height 48.0 Points
Month of: [INV LDH]
[Product] [Qty Gty] [Value]
Body: Header; Objects 3, Height 14.0 Points
Month Footer. Objects 4 Height 48.0 Points
Monthly Summ [M(TOTAL)] [MODULA]
Summary, Objects 3, Height 36.0 Points
[Summary] [TOTAL] [DOLLAR]

Product Sales Summary

Month of: Nov, 1992

Product	Quantity	Value
Database	62	$26,137.00
Spreadsheet	59	$21,944.00
Monthly Summary	121	$47,081.00

Month of: Dec. 1992

Product	Quantity	Value
Database	62	$74,962.00
Spreadsheet	53	$19,875.00
Monthly Summary	115	$44,737.00

Summary 236 $91,740.00

To use CodeReporter, simply draw your report, then include it in any program you write. Call (071) 833-1022 now for your FREE working model of CodeReporter.

New - Design complex reports in just minutes...

Our new CodeReporter takes the painstaking work out of reports. Now simply design and draw reports interactively under Windows 3.1, then print or display them from any DOS, Windows or UNIX application.

HELD OVER! - FREE CodeReporter
Order CodeBase 5 before Dec 30, 1993
and receive CodeReporter for free!

Distributed & supported by

SEQUITER SOFTWARE UK LTD
106-108 POWIS ST
LONDON SE18 6LU
TEL: 081 317 4321

CodeBase 5.0
The C.C++ Library for DataBase Management

Call Now
071 833 1022

System Science
3-5 Cynthia Street, London, England, N1 9JF
Tele: +44 71 833 1022 **Fax:** +44 71 837 6411

➤ CIRCLE NO. 138

One possible approach would be to move the allocation of the memory to a separate function and to make this function virtual, so that the constructor for Motor calls the version overridden in IndustrialMotor. Unfortunately, this approach simply doesn't work. If the compiler works according to the specification in the *Annotated Reference Manual*, it will ignore the virtual-ness of the function and simply generate a call to Motor's version of the function. This strange behaviour ensures that the derived instance does not receive function calls before it has been constructed.

A more successful approach is to define an additional protected constructor for Motor, which does not allocate memory. Then we can use the constructor initialisation list for IndustrialMotor to call that constructor rather than any other. The parameter for this protected constructor is the already defined type MotorImplementation*, which should not be a parameter to any publicly visible constructor for Motor. As it stands, the IndustrialMotor class will correctly call functions defined by Motor and implemented by MotorImplementation. However it requires some more work before we can define new functions. The pointer in Motor can be protected, rather than private, but it is still of type MotorImplementation*. So if we are to use it to call functions specific to IndustrialMotorImplementation we must cast the pointer. We can define an inline function to do this cast.

There's still another potential problem; the destructor. We cannot prevent the destructor being called for the Motor instance, so it will call delete for the base class as well as for a MotorImplementation - even though it is actually an instance of the derived class. As usual, the solution is to declare the destructor for the MotorImplementation class as virtual. There is no need to define a separate destructor for the IndustrialMotor class. The resulting code in the public header files might look similar to that given in Figure 8.

Note that there is no reason to make functions virtual in Motor or IndustrialMotor, nor is there a need to redefine these functions in derived classes. Instead, we can make the functions virtual in the implementation classes as shown in Figure 9.

With this 'boiler plate' code for each class, inheritance will work correctly. This inheritance doesn't add any further overheads or complexity for the compiler.

```
// Forward declaration
class MotorImplementation;

class Motor
{
public:
  Motor( char* );
  ~Motor();
  void start();
  void stop();
  void setSpeed( float );
protected:
  Motor (MotorImplementation* x) :
    p( x ) {};
  MotorImplementation * p;
};
class IndustrialMotorImplementation;
class IndustrialMotor
  : public Motor
{
public:
  IndustrialMotor( char* );
  float getSpeed();
protected:
  IndustrialMotor
  (
    IndustrialMotorImplementation*
  );
private:
  // Casting function
  IndustrialMotorImplementation * asIMI()
  {
    return
      (IndustrialMotorImplementation*) p;
  }
};
```

Figure 8 - An example cheshire Cat virtual destructor

```
class MotorImplementation
{
  friend class Motor;
protected:
  MotorImplementation(char*);
  // Ensure constructors are virtual
  ~MotorImplementation() {}
  // This function must now be virtual
  virtual void start();
  // so make the rest for consistency
  virtual void stop();
  virtual void setSpeed( float );
private:
  // Private functions and data go here...
};
class IndustrialMotorImplementation :
  public MotorImplementation
{
  friend class IndustrialMotor;
protected:
  IndustrialMotorImplementation(char*);
  // Re-implemented function.
  void start();
  // Newly defined function.
  virtual float getSpeed();
};
Motor::Motor( char* s )
  : p(new MotorImplementation(s)) {}

Motor::~Motor()     { delete p; }
void Motor::start() { p->start(); }
void Motor::stop()  { p->stop(); }
void Motor::setSpeed( float s )
{
  p->setSpeed(s);
}
IndustrialMotor::IndustrialMotor(char* name)
  : Motor(new IndustrialMotorImplementation
              (
                name
              )
          ) {}
IndustrialMotor::IndustrialMotor
(
  IndustrialMotorImplementation* x
)
  : Motor( x ) {};

float IndustrialMotor::getSpeed()
{
  return asIMI()->getSpeed();
}
```

Figure 9 - Implemening a cheshire Cat virtual destuctor

Conclusion

So, in practice, both abstract interface and Cheshire Cat classes can separate the interface of a class from its implementation, and both approaches can handle inheritance. However each has its drawbacks. Cheshire Cat classes:

- Require rather more "boiler-plate" code for each class and function defined.
- Have a (usually insignificant) processing overhead on each function call, and a space overhead on each instance created.

By comparison, Abstract Interface classes:

- Cannot be created in the normal way, using code.
- Make heavy demands on the compiler implementation.

Thus for a class library one might chose the Cheshire Cat approach; for a large, time-critical, application using a very reliable compiler one might choose abstract interface classes; a small application would probably use neither. The choice will depend, therefore, on the demands of your particular application, and the support provided by the compiler you are using.

Charles Weir is a consultant at Object Designers Ltd. He can be contacted on 0279 755396 or by email as cweir@cix.compulink.co.uk.

"WE USED A VCS BECAUSE WE WERE TOLD TO, NOW WE USE 9k BECAUSE WE WANT TO!"

Why? Because 9k's Project Oriented and Integrated Approach make configuration control so much easier than other VCS's we could mention.

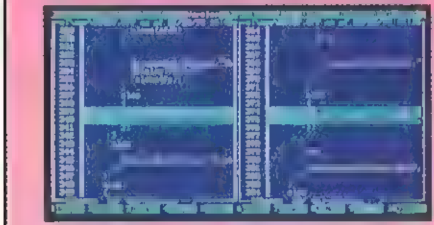
- Inventory & change control
- Roll-back recovery
- Version and file locking

9k's file selection and highlighting shows you "who has what" and which files have changed.

> "9k reduces the complexity of using version control software..."
> DATABASED ADVISOR

The integration of the software inventory with version control means that you don't need to remember the set of files that constitute a version nor which revision of each file belongs to that set.

Use 9k's own menu or call 9k with a macro or batch command, whichever way you do it, 9k fits in with the way you work and is a joy to use.



9k shows the changes in it's unique split-screen display.

- File, version and change descriptions
- Access control
- Branching
- Archive file compression

9k is language independent, but additional dBase file support makes it particularly useful to Xbase developers.

> "This product offers a combination of power and ease of use...The cost of 9k is economic and it has a reasonable claim to be version control for everyone."
> PROGRAMNOW

NEW 9k Version 2.00 available from just £149.00 with a 30 day money-back guarantee.

Forté Software Tools Limited
Hunsdon House
Walford
Ross-on-Wye HR9 5SB.
Tel 0989 767656
Fax 0989 769091

(9k is so called because it represents the international quality standard ISO 9000)

Version Control without Tears

Call QBS NOW on 081-994-4842 for further information or demo disk. QBS Software Ltd.
10 Barley Mow Passage, Chiswick, London W4 4PH.

(c) 1993 Forté Software Tools Ltd, All other trademarks acknowledged

➤ CIRCLE NO. 139

NEW FROM MICROSOFT PRESS

MICROSOFT WINDOWS NT RESOURCE KIT

Windows NT Resource Guide
Microsoft WINDOWS NT RESOURCE KIT
Windows NT Messages
Microsoft WINDOWS NT RESOURCE KIT
Optimizing Windows NT
Russ Blake
Microsoft WINDOWS NT RESOURCE KIT

Complete Technical Information and Tools for the Support Professional

This exclusive three-volume Microsoft collection is a comprehensive source of technical information needed to troubleshoot and optimize Windows NT and the Windows NT Advanced Server. Its purpose is to provide the technical information and tools necessary for self-support of installations of Windows NT in your company. If you want to become an expert user of Windows NT, the Microsoft Windows NT Resource Kit is for you. The three-volume boxed set includes seven 3.5" disks and one CD-ROM containing all the disk-based utilities plus tools and utilities for MIPS - based computers.

Three-Volume Boxed Set

ISBN 1-55615-602-2 £89.70+VAT

Contents include:

Volume 1: *Windows NT Resource Guide*
Super-secret, trouble-shooting, utilities and accessories for power users of Windows NT.
950 pages. Four 3.5" disks. ISBN 1-55615-598-0 £38.26+VAT

Volume 2: *Windows NT Messages*
Alphabetical reference and electronic database containing all Windows NT messages - in plain English.
600 pages. Two 3.5" disks .ISBN 1-55615-600-6 £27.66+VAT

Volume 3: *Optimizing Windows NT*
The complete guide to maximising the capacity and speed of Windows NT on the desktop or network.
350 pages. One 3.5" disk. ISBN 1-55615-619-7 £23.79+VAT

Each title also available separately.
Complete contents listing available on request. **Available October 93**

Advanced Windows NT

Jeffrey M Richter

The developer's guide to the Win32 Application Programming Interface, ideal for those making the transition from Windows programming.
Each chapter attacks a specific topic of Windows NT programming, explaining how to exploit the feature or mechanism in their program.

Contents include:

- Heaps
- Virtual Memory Management
- Memory-mapped files
- Processes and threads
- Messages and input states
- Exploited DLLs under Windows NT
- Building debugging tools
- Manipulating the three file systems that Windows NT supports (FAT, HPFS and NTFS)
- Equipping applications with structured exception handling
- Designing language-independent applications for localization

660 pages ISBN 1-55615-567-0 £34.95+VAT **Available November 93**

Available from all good bookshops. For mail order service call penguin Direct on 081 899 4036, or to request a catalogue ring 071 416 3000.

➤ CIRCLE NO. 140

Windows NT - Moving up a gear

Now that you can finally buy Windows NT, the race is on to get applications up and running - Mike Warriner describes some of the new features.

It's nearly six years since I first used Microsoft's new vision for the future. Since then, this vision has swayed from Windows to OS/2 and back to Windows. Finally, six years on, I truly believe that Microsoft has set the foundation for the API that we can program to for the next half decade. That's a pretty tough challenge and I expect that you UNIX gurus will disagree - of course, you may be right. What I *do* believe is that in five years time, your (well written) Win32 program will still compile and run on whatever computers we have then. Try that with UNIX!

Win32 is this foundation. It is the API that Microsoft wants you to use to program under Windows NT and all future Microsoft operating systems. Being primarily an API, there is no reason why it should not be available on everything from the smallest PDA to the largest mainframe.

Within the length of this article, I hope to be able to introduce what I feel are the most significant new features of Windows NT. Hopefully, by the end, you will understand enough to be able to get to work.

I shall refer to the API as the Win32 API throughout this article - this is the full feature API available under NT3.1. Various features (such as security or Unicode) may not be available on other Win32xx implementations.

The future

Microsoft has announced plans to port NT to as many machines as possible, encompassing both big-endian and little-endian: as such, the documentation goes into great detail about the compatibility problems that it expects programmers to encounter if they ignore a few ground rules. If you really believe that anyone will ever try porting NT to a non big-endian machine, you should read these parts of the manuals carefully - the manual discusses problems such as the byte order of Unicode being reversed and binary data being confused.

```
 / Remove this define to compile on
// non-unicode Win32 system

#define UNICODE

#include <stdio.h>
#include <stdlib.h>
#include <windows.h>

#ifdef UNICODE

#define NLSCHAR         wchar_t
#define NLS LIT(x)      L##x

// I've defined NLS_CHARLIT to be like
// this because I feel that
// 65535 == NLS_CHARLIT('\FFFF')
// should be true. Windows NT believes this
// to be false.

#define NLS_CHARLIT(x)  (NLSCHAR)L##x
#define NlsPrintf       wprintf
#define NlsSprintf      swprintf

#else

#define NLSCHAR         char
#define NLS_LIT(x)      x
#define NLS CHARLIT(x)  (NLSCHAR)x
#define NlsPrintf       printf
#define NlsSprintf      sprintf


#endif

int main()
{
   NLSCHAR nszMyString[100];
   NlsSprintf(nszMyString, NLS_LIT("Mike"));
   NlsPrintf(NLS_LIT("My name is %s\n"),
                   nszMyString);
   return 0;
}
```

Figure 1 - Example of ASCII/unicode portabilitly

Given announcements from Sun and various other manufacturers that they will be redesigning their chips to support NT, I can't see this ever being a problem - other problems of compatibility are more immediately important, in particular:

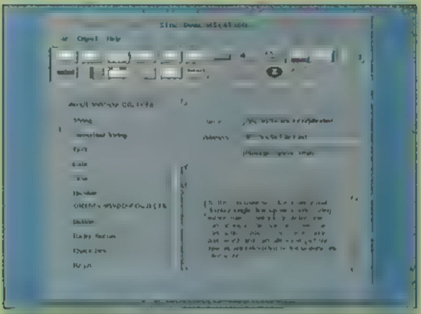
- **Makefiles** Although the source code should be identical, compilers are free to take completely different options on different platforms. Microsoft provides a generic makefile called NTWIN32.MAK to solve this problem. This makefile contains all the logic required to supply the correct compile and link options to build applications on all possible platforms.

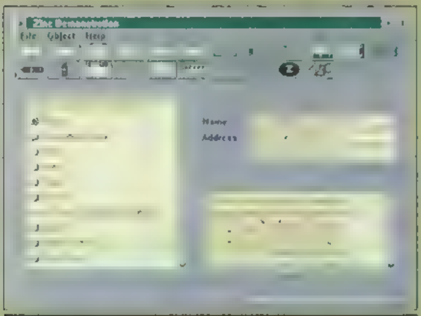
- **Using undocumented features** Surprisingly, there is still ample opportunity for undocumented features in the NT environment. Some will be the same on other machines, some won't. One particular example I found is the structures associated with Critical sections: we were using one in an internal counter to allow us to free memory when the usage count hit zero. On the Alpha the counter behaved slightly differently. In this case we were probably at fault, but you get the idea.

Unicode

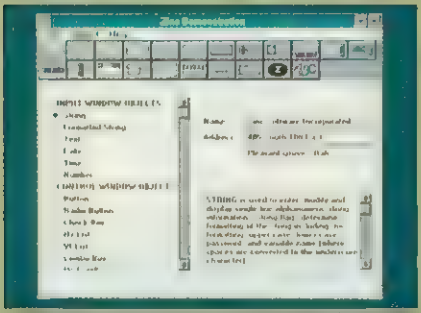
By providing a 16-bit character set that has defined unique values for each character available, applications no longer have to worry about which code-page is needed to display the Greek alpha, or whether the current code page actually has an 'ö' in it. I will show the extensions that Microsoft has made to the C syntax to use Unicode, OBJECT-ORIENTED · MULTIPLATFORM · SINGLE-SOURCE · C++

THINK ZINC

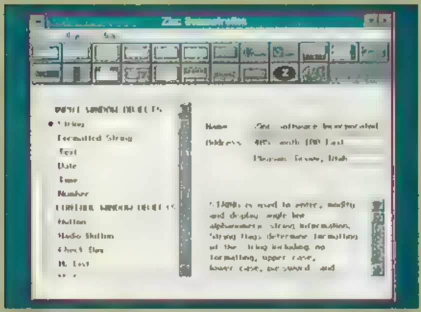

UNIX MOTIF



OS / 2 2.0



MS WINDOWS & WINDOWS NT



DOS GRAPHICS

DOS TEXT

ZINC™ APPLICATION FRAMEWORK™ 3.5

Multiplatform Flexibility From One Application Framework.

FLEXIBILITY *is an essential component in software design. Your development tools should allow you to easily incorporate new features and technologies into your applications without rewriting all of your code. That's a tall order if your development tools are designed like a straight-jacket. Zinc Application Framework 3.5 and Zinc Designer™ give you flexible and extendible support for Microsoft Windows, Windows NT, OS/2 2.0, UNIX Motif, DOS Graphics and DOS Text. And Zinc does it with ONE set of source code. Zinc's multiplatform, object-oriented architecture won't confine your application development options today... or tomorrow.*

FOR A FREE ZINC DEMO KIT, CALL US TOLL FREE TODAY AT 1.800.638.8665. IN EUROPE CALL +44 (0)81 855 9918

Z i n c

ZINC SOFTWARE INCORPORATED, 405 SOUTH 100 EAST, 2ND FLOOR, PLEASANT GROVE, UTAH 84062, TEL 801 785 8900, FAX 801 785 8996, BBS 801 785 8997

EUROPE: ZINC SOFTWARE (UK) LIMITED 58-60 BERESFORD STREET, LONDON SE18 6BG, TEL +44 (0) 81 855 9918, FAX +44 (0) 81 316 7778, BBS +44 (0) 81 317 2310

➤ CIRCLE NO. 141

and how an application can be written to use Unicode under NT, but still work under other Win32 platforms (such as Win32s) that do not support it.

Microsoft has extended the C/C++ languages to accept Unicode character strings and characters. This involves using the prefix 'L' in front of all Unicode strings. Internally, the compiler generates 16 bits of data for each character instead of the usual 8 - the new character type (`wchar_t`) is actually an unsigned short.

```
L"This string is in Unicode"
L'X'
L'\xFFFF'
```

Unfortunately, Microsoft has made it clear that it will not support Unicode under all future variations of the Win32 API. It is already not supported under Win32s. If you think that your application may ever need to run on these systems, and it will, you should use a set of macros to encapsulate the Unicode extensions to allow you to write more portable code.

Figure 1 shows an example program that uses a methodology similar to the one we use at Intelligent Environments for our code. The `#defines` are in a header file that program modules can include to guarantee compatibility with both Unicode and non-Unicode systems without altering any of the code.

Microsoft has made the Win32 API completely transparent to Unicode: except for the conversion functions, all other functions are actually mapped to use the correct version depending on the `UNICODE` macro. Unfortunately, it has not taken this approach with the C libraries, and appears not to have documented the resulting function names!

As a result, if you use Win32 functions entirely, the Unicode/non-Unicode function name translation is automatic. However, if you need to use C library functions, you will have to scan the header files to find the Unicode name for your ASCII function: examples include `wprintf`, `wcslen`, and `fputw`.

The Registry

Under Win32, the registry is a completely protected pseudo file system designed to hold textual and binary configuration information for every application on the system. There are databases for the local machine, and for each user on the machine, allowing individual users to customise their applications independently.

Under Windows NT, the operating system has completely removed all dependency on textual configuration, moving all this configuration information into a registry database. So CONFIG.SYS, AUTOEXEC.BAT, WIN.INI, SYSTEM.INI and all the other .INI files have been removed. Various reasons have been given, perhaps the most important ones are:

- Security of .ini information. Casual users can't prevent the operating system from working.
- Network aware APIs allow system administrators to set registry on a remote user's machine.
- Separation of user and system configuration. Since NT knows which user is using the machine, it can provide different configuration information for each user.

For example, to read the file name of the current Windows wallpaper, you simply read the tag `HKEY_CURRENT_USER\Control Panel\Desktop\Pattern` from the registry. Many other pieces of information exist which are listed in Chapter 52 of the reference manual.

Microsoft recommends that user applications store their information in the `HKEY_CURRENT_USER\Software` key with the form: `\CompanyName\ApplicationName\ApplicationVersion\SubKey\...` This information should be machine independent (it could be accessed from different machines on the network), and generally under a couple of kilobytes per item. In fact, the limit is 1 Mb per item plus your disk space for the entire registry. For example, we store the `X` position of the main window of our current test rig in the key:

```
HKEY_CURRENT_USER\Software\IE\
TestRig\0.5\Windows\
MainWindow\Position\X
```

Microsoft supplies a new API set to handle the registry, prefixed by '`Reg`'. Keys must be opened before use, and closed afterwards like normal files. While open, single or multiple keys can be read or written into memory or to disk using the `RegQueryValueEx`, `RegSetValEx`, `RegLoadKey` and `RegSaveKey` functions respectively.

Threads

Threads are *the* way of achieving multi-tasking within a single program. They allow different parts of a single program to execute independently, but with access to the same variables, files and resources as each other. Contrast this to the UNIX 'fork' style multitasking where processes cannot get access to each others variables, and resources are only shared from creation.

Examples of threads can be seen everywhere when using NT, the most visible is on the File dialog. With a slow network, the network drive letters appear several seconds after the dialog appears, but the dialog is usable instantly. Other examples often cited are background printing, saving, repaginating, redrawing. The list goes on...

```
//
// Simple program to show usage of
// VirtualAlloc and exception handling

#include <stdio.h>
#include <windows.h>

//
// Protected string length routine.
//
int SSLEN(char *s)
{
  int count = 0;

  try
  {
      while (*s++ != '\0')
         count++;
      return count;
  }
  except
  (
    GetExceptionCode() ==
      EXCEPTION_ACCESS_VIOLATION ?
        EXCEPTION_EXECUTE_HANDLER :
        EXCEPTION_CONTINUE_SEARCH
  )
  {
    printf("Exception occured\n");
    return count;
  }
}

int main(void)
{

  printf("Strlen %d\n",
    SSLEN("hello")); // prints 5
  printf("Strlen %d\n",
    SSLEN(NULL));    // prints 0

  return 0;
}
```

Figure 2 - Protected string length routine

Clipper Has It All.

Attention dBase, FoxPro, Paradox and Access Users: Upgrade To CA-Clipper® 5.2 For Only £99 And Choose Any One Of These Products For Only £50! Offer Good For Users Of Any Competitive DBMS – Call Today!

Break through the 640k barrier with the ultimate extended memory manager! Increase directly addressable memory from 640k to 16 megs.

Over 550 time-saving, problem-solving functions in one high performance package. Raise your productivity with this valuable library.

dBASE apps come alive with new CA-dBFast 2.0 – the first dBASE-compatible database and language for Windows.

CA-Clipper 5.2: The Complete, Professional Programming Environment.

CA-Clipper® 5.2 is a robust language, an efficient linker, a flexible preprocessor and a high-performance compiler. It includes: an editor, debugger and make utility for creating PC- and LAN-based applications.

Break Through The 640k Barrier With New CA-Clipper/ExoSpace!

CA-Clipper®/ExoSpace™ increases directly addressable memory by up to 2,500% – from 640k to 16 megabytes. Eliminate virtual memory swapping, greatly improve performance and run your Clipper applications in protected mode. Existing 5.2 users can get CA-Clipper/ExoSpace for only **£99!**

New RDDs For FoxPro, Paradox and dBASE IV.

PC Magazine Editor's Choice for Xbase Development

DBMS Reader's Choice winner for client/server development system

Replaceable database drivers for all the most popular development systems. Plus you can customise CA-Clipper with user-defined commands and functions. And seamlessly integrate modules from languages such as C, Assembly, dBASE and Pascal. There are no runtime fees, no additional licences, no LanPaks.

So what are you waiting for?

Call right now and upgrade to the new standard in Xbase development.

Feature	CA-Clipper	FoxPro	dBASE IV
Lexical Variable Scoping	YES	NO	NO
User-defined Commands	YES	NO	NO
Code Blocks	YES	NO	NO
Nested Multidimensional Arrays	YES	NO	NO
Predefined Object Classes	YES	NO	NO
Variables Modifiable In Debugger	YES	NO	NO
Customizable Reactive Error Handling	YES	NO	NO
Reads and Writes FoxPro (.CDX) Files	YES	YES	NO
Reads and Writes dBASE IV (.MDX) Files	YES	NO	YES
Reads and Writes Paradox (.DB/.PX) Files	YES	NO	NO

Comparison based on FoxPro Version 2.5 for DOS and dBASE IV Version 1.5.

1993 DATA BASED ADVISOR Readers Choice Award

DBA readers rated CA-Clipper 5.01 Best Development Language, Best Xbase Compiler and Best Development Product.

COMPUTER ASSOCIATES®
Software superior by design.

Call The CA-Clipper Upgrade Desk On **0753 577733.**
Offer good for a limited time only, so call today!

New CA-Clipper® 5.2

© Computer Associates Plc., Computer Associates House, 183-187 Bath Road, Slough, Berkshire SL14AA. All product names referenced herein are trademarks of their respective companies.

➤ CIRCLE NO. 142

Since threads are not a new concept, I won't go into too much detail with their usage under NT. The most important functions are `CreateThread` and `SetThreadPriority`.

Threads are created with their own stacks but in the same memory space as the creating thread. Unless otherwise specified, they can start execution immediately.

Under NT, each thread can have one of four basic priorities set for it: `IDLE`, `NORMAL`, `HIGH` and `REALTIME`. The default is `NORMAL`. Impressively, `REALTIME` executes at a priority above most of the operating system, allowing you to completely halt NT by running a CPU intensive application at this priority.

The biggest problems with threads occur when novice programmers decide that they can easily implement their application by starting 20 threads and running every single window/operation on a different thread. I would liken this to running a complex building project by starting building the roof, floor, ceilings, wiring and painting all at the same time, and expecting that somehow they will be done in the right order.

Synchronising more than only a couple of threads becomes extremely complex and unreliable. The moment you find that you can't answer the question: 'What's going on now?' without a debugger, you have too many threads. This problem is very real. At Intelligent Environments we now restrict our programs to a few threads after we tried using six on initial versions. We experienced bizarre lockups as several threads tried to access resources that should have been available, but were occasionally locked if two threads executed in a different order.

Synchronisation

Win32 provides four different synchronisation methods, split into two distinct groups: *critical sections* and *objects*.

The simplest method of synchronisation is the concept of critical sections. Win32 provides these within a process.

```
//
// Simple program to show usage of
// VirtualAlloc and exception handling
#include <stdio.h>
#include <stdlib.h>
#include <windows.h>


// For this example, I commit from Address
// to Address+255. This will guarentee that
// the rest of the strcpy will succeed.

// I could just as well have only committed
// one byte, and waited for the strcpy to
// fail on the next byte before continuing.

INT MyExceptionFilter
(
  DWORD dwCode,
  char *Address
)
{
  if (dwCode != EXCEPTION_ACCESS_VIOLATION)
  {
    printf("Unknown exception\n");
    return EXCEPTION_EXECUTE_HANDLER;
  }

  printf("Allocating new page\n");

  if (VirtualAlloc( Address, 255,
                    MEM COMMIT,
                    PAGE_READWRITE))
    return EXCEPTION_CONTINUE_EXECUTION;
  else
    return EXCEPTION_EXECUTE HANDLER;
}

void SaveAString
(
  char *Base,
  int x,
  int y,
  char *AString
)
{
  char *Address;
  //
  // Find start address for string
  // (yuck - pointer arithmetic!)
  //
  Address = Base + (x + y*1000) * 255;

  try
  {
    strcpy(Address, AString);
    printf("Strcpy is ok\n");
  }
  except (MyExceptionFilter(
              GetExceptionCode(), Address))
  {
    printf("Exception not handled\n");
    ExitProcess(GetLastError());
  }
}

int main()
{
  char *RBA;
  int  i;

  RBA = VirtualAlloc(NULL,
                     1000*1000*255,
                     MEM_RESERVE,
                     PAGE_NOACCESS);
  //
  if (RBA != NULL)
  {
    for (i=0; i<10; i++)
    {
      SaveAString(RBA,
                  rand()%1000,
                  rand()%1000,
                  "Hello world!");
    }
    VirtualFree(RBA, 0, MEM_RELEASE);
  }
  return 0;
}
```

Figure 3 - Using VirtualAlloc and exception handling

A thread requests entry to a critical section and blocks until entry is obtained. Once obtained, all other requests to entry from the same thread are accepted and increase a lock count. All requests from other threads block until either the critical section is left, or the owning thread dies.

More sophisticated synchronisation exists in the form of synchronisation objects. Currently three types of objects exist: *Event objects*, *Mutex objects* and *Semaphore objects*. Commands exist to wait for an object(s) to change state, to change an object's state, and to create objects with specified names. Objects can be shared between processes, even over a network, allowing synchronisation of shared memory and files.

The various objects have specific uses, some of which are:

- **Event** Notifies one or more threads that an event has occurred. Such events could be a program starting, a network packet arriving or a signal to several threads that their parent has completed initialisation and that they can start executing.
- **Mutex** Like the proverbial 'cookie', one thread in the system owns the Mutex object, all others can request ownership. Useful for sharing a fixed resource such as shared memory.
- **Semaphore** Like a Mutex object, except that the semaphore maintains an ownership count so that more than one object can access the resource simultaneously.

Unlike the sometimes maligned OS/2 mutex semaphores, if the owner of a Windows NT synchronisation object dies, the object goes into an 'Abandoned' state. This effectively releases ownership and lets the next owner know that the object may be in an undefined state.

Exceptions

Modern business applications quite simply cannot be allowed to fail. Now that software is running most businesses, controlling billions of dollars of financial transactions every day and flying planes throughout the globe, a simple system crash is completely devastating. We must set ourselves the goal of producing

New Version 3!

If You're Wondering About the Latest In Database Developer Technology...

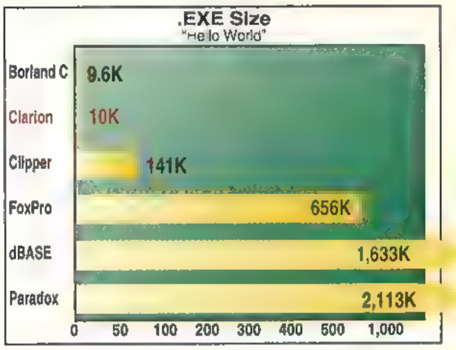
This Should Bring You Up To Speed!

Faster, More Powerful, More Versatile • The Clarion Database Developer™ Version3

We know it's hard to believe, but at last, everything you've ever wanted in a database development platform is available now in the NEW Clarion Database Developer, Version 3.

C Language Performance

Our optimizing compiler and "smart" linker deliver run time performance that matches the best C compilers. "Hello World" takes only 10K. Benchmarks like the "Sieve of Eratosthenes" run 200 times faster than FoxPro and Clipper. The compiler produces standard .OBJs that can be linked with other languages and debugged with our own source level debugger. (See charts)

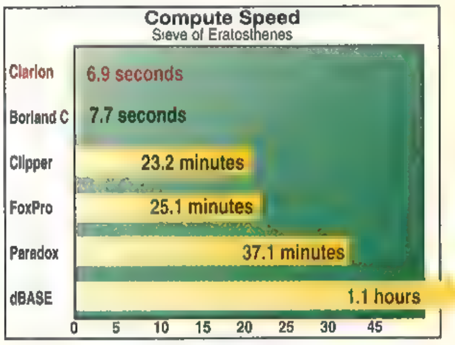


Leading Edge Memory Management

Break the DOS 640K barrier with our automatic overlay system. Segment your application into DOS DLLs that can be distributed separately. Or run your application in protected mode with our optional royalty-free hardware assisted DOS extender-the ultimate memory manager.

Awesome Video Effects

Create SAA/CUA compliant applications with pull-down menus and attractive graphic check boxes, radio buttons, and smooth mouse cursor. Place graphic images of any size on your text screens. Use any text mode: 80x25 to 132x44. Design virtual screens up to 16 times larger than normal. Open exploding or falling windows with see-through areas and transparent shadows.



Replaceable Database Drivers

A universal data dictionary works with Clarion database drivers to produce applications for any database. Xbase, Paradox, Btrieve, c-tree, ASCII, and Clarion drivers are standard. SQL databases are available at an additional cost. Changing a single parameter ports your application to a different database.

Interactive Visual Design

Assemble your application from our catalog of visual procedure templates then "paint" in your custom screen and report layouts. Embed your own source wherever it is needed. Then we generate commented pre-tested source code for you.

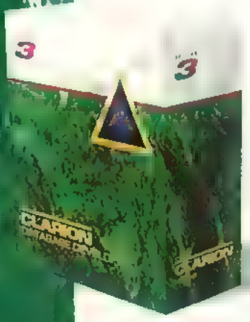
Optimistic Concurrency

Clarion applications support optimistic concurrency to maximize your multi-user throughput and referential integrity to safeguard your data. These strategies work on any database concurrently with any application.

No Royalties or Runtimes

Your applications compile into .EXE files. No royalty payments, no run-time modules, no copyright notices, and no LAN Packs or Bump disks.

Product Comparison	dBASE	FoxPro	Paradox	Clipper	CLARION	Borland C
Visual Design	Partial	Partial	Partial	No	Complete	No
Data Dictionary	No	No	No	No	Yes	No
Database Support (Read/Write "in place")	Xbase	Xbase	Paradox	Xbase Paradox	Xbase, Paradox Btrieve, SQL, etc	None
Royalties or Run-times	Yes	Yes	Yes	No	No	No
Network Support	w/LAN Pack	w/LAN Ver.	w/LAN Pack	Unlimited	Unlimited	No
Memory Models	Overlay	Overlay Protected	Overlay	Static, Overlay	Static, Overlay DLL Protected	Static, Overlay



Introducing the New Clarion Database Developer Version 3

CLARIONSOFTWARE

.EXE 11/93

Yes, I Want To Be Brought Up To Speed!

Please send me information about Clarion NOW.

Name: ______________________

Company: ______________________

Address: ______________________

____

______________________ Post Code: ______________________

Tel: ______________________ Fax: ______________________

For a FREE Full Colour Brochure, please post or fax this coupon to:
Clarion Software (Europe) Ltd., 3 The Mansards,
Tavistock Street, Bedford, MK40 2RX, England.
Fax: +44 (0)234 217094 or Call: +44 (0)234 267500

➤ CIRCLE NO. 143

completely secure software: not bug free, more bug secure.

Windows NT's nested structured exception handling provides an excellent tool to help meet this goal. With this, and the extensions to the C language, we should be able to banish the UAE to the dustbin forever.

Like most of NT, the exception handling is built around the concept of objects. These can be any module of code, a function, library, DLL or the operating system. We can view each of these objects as being packages of 'methods', each method supposedly having a defined set of entry conditions and exit results. If any of these entry conditions is not met, the program is in error, and will cause some kind of exception to be signalled. In the case of a bad parameter, a simple return code may suffice. We have a memory access violation, an `ACCESS_VIOLATION` exception will be generated.

Before Win32, these exceptions were handled, if possible, by the use of `setjmp/longjmp` or `Catch/Throw` pairs. CPU faults were commonly handled by exception handlers (such as under OS/2), which were usually able to terminate the program gently, but rarely able to recover the execution flow from a convenient position.

Under Win32, Microsoft has provided a mechanism that means that no one can really make any excuse for not implementing exception handling and recovery. The `try-except` construct allows programs to execute code safe in the knowledge that it can handle any exceptions that the code generates. Since the mechanism is nested, it is possible to handle only a subset of exceptions, and to pass the rest up to another exception handler.

A simple example is that of `strlen`. The standard string length function will give an `ACCESS_VIOLATION` error if the pointer is invalid, or the string is not zero-terminated. Figure 2 shows a simple `strlen` function that will handle this exception without causing the program to terminate. A more sophisticated routine would log the error, or try to cure the problem (perhaps providing the zero termination).

Further sophistication can be added by using the `try-finally` construct, which guarantees that a piece of code will be executed whether or not an exception occurs. The addition of the `RaiseException` function allows the programmer to generate user exceptions, perhaps allowing a new `assert` macro to be generated that simply causes an exception instead of exiting the program if the condition is not met.

Memory

Under Win32, each process has it's own virtual address space of 2 GB. Win32 has merged most of the old Windows APIs together, and added a few new ones to support sparse memory allocation. Among other functions, the new virtual memory functions offer the ability to:

- Reserve a range of a process' virtual memory space. This does not allocate any physical storage for the memory, but allows an application to reserve the maximum size of a structure and allocate physical storage only when needed.
- Commit memory. This allocates the physical storage for the memory.
- Specify access criteria for committed memory. Access can be Read-Only, Read/Write or no access.
- Allow access to memory for other processes (shared memory).
- Free committed/reserved memory, uncommit memory, lock and unlock memory and query information about memory blocks.

Along with the exception handling code, sparse memory gives an extremely simple method of allocating memory on demand. For example, to allocate memory for a sparse array of 1000 by 1000 strings, each of up to 255 characters. There are three possible ways to go about accessing such an array. First we could work out a complex algorithm, perhaps using linked lists, to provide an API to access this array. Alternatively we may try to locate a machine with 255 MB of free virtual memory (disk/ram) and type `malloc(1000*1000*255)`. Or we may chose to reserve 255 MB of virtual memory, and only commit each block when the block is required.

For this example, we would all probably opt for the first solution, but see Figure 3 for an example of how to implement the third. It's *much* easier than any other solution, something that should mean that the code is much more reliable and easier to debug. Also note the simplicity of the actual storage algorithm. I use `strcpy`, but any function that writes memory to this reserved region would also work - I don't care where it actually outputs the data, any access violation within the reserved range will be automatically resolved.

Aside from sparse memory, Windows NT also fully supports shared memory between processes through the use of file mapping. This API gives two sets of functionality, both are unique to NT and incredibly useful. The first is simple shared memory. One process creates a file mapping by using the `CreateFileMapping` API and specifying the name of the mapping object. Other processes can then open it using the `OpenFileMapping` API. The second use is in that the file can be either a virtual file within the swapping file, or a physical file on the disk. As well as shared memory, the file mapping API lets you map a file into the address space, giving complete random access to a disk file, but not requiring that you code any buffering logic, load or save logic.

Conclusion

Windows NT is poised to set the standards for our programming environments for the next half-decade. Using the features I have described will give your programs an edge over the competition, in reliability, portability and power. For the first time in many years, PCs have a more powerful and more robust API than UNIX - use it!

.EXE

Mike Warriner is currently involved in developing software products running on both OS/2 and Windows NT for Intelligent Environments. He can be contacted by email as mike@castout.demon.co.uk.

References

The Unicode Standard: Worldwide Character Encoding, Version 1.0,

Volumes 1 and 2, Addison-Wesley Publishing Company: 1991, 1992, ISBN 0201567881.

Visual Basic™ goes Client/Server!

NEW!

Q+E MultiLink•VB

links Visual Basic to all major databases

With Pioneer Software's new Q+E MultiLink•VB, you can create powerful Visual Basic database applications that access data in over 20 PC and SQL database formats. Now you can build true client/server database applications and distribute them *Royalty Free.*

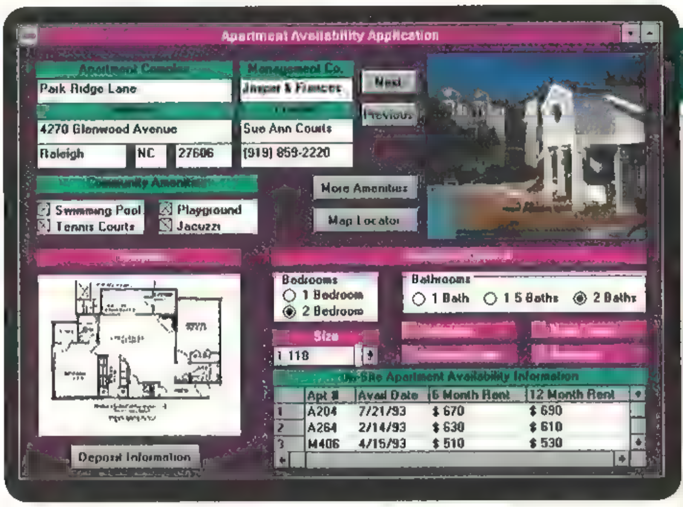
Q+E MultiLink•VB is the fastest way to build database applications—even if you need to access data from multiple formats. Use it to display data residing in several different databases together on the same form. Because Q+E MultiLink•VB works the same way for all database formats, you don't have to learn a different interface for each database. It's not only easy to learn—it's easy to use. You can even define sophisticated queries without knowing SQL.

Create powerful database applications

Q+E MultiLink•VB includes several graphical development tools so you can quickly deliver powerful database applications. Take advantage of the database custom controls, the Database Manager, and the Query Builder to transform Visual Basic into a complete client/server development tool.

Custom controls eliminate code

Custom controls added to the toolbar enable you to design your applications without writing code. In Q+E MultiLink•VB, you'll find the same controls included in our award-winning database for Visual Basic, Q+E Database/VB, and more.



Apartment Availability Application

Apartment Complex: Park Ridge Lane

Management Co.: Jasper & Frances

Next

Previous

4270 Glenwood Avenue

Raleigh | NC | 27606

Sue Ann Courts

(919) 859-2220

Community Amenities

☑ Swimming Pool ☒ Tennis Courts ☒ Playground ☒ Jacuzzi

More Amenities

Map Locator

Bedrooms: ○ 1 Bedroom ● 2 Bedroom

Bathrooms: ○ 1 Bath ○ 1.5 Baths ● 2 Baths

Size: 1 118

Deposit Information

On-Site Apartment Availability Information

	Apt #	Avail Date	6 Month Rent	12 Month Rent
1	A204	7/21/93	$ 670	$ 690
2	A264	2/14/93	$ 630	$ 610
3	M406	4/15/93	$ 510	$ 530

Quickly create powerful Visual Basic database applications.

Database management made simple

Use the Database Manager to create, view, modify, and delete databases from within Q+E MultiLink•VB. Design a database table and specify the field names, data types, and field lengths—then link the table to your application.

Build sophisticated queries

Retrieve data from more than 20 database formats with our highly acclaimed Query Builder. The Query Builder leads you through the creation of SQL Select statements in a few simple steps. Just point and click to define the entire query before sending the request to the database.

Familiar with Q+E Database/VB?

New Q+E MultiLink•VB provides a seamless transition for Q+E Database/VB applications. In a matter of minutes, you can use Q+E MultiLink•VB to link applications built with our first Visual Basic product to all major databases. Call for details on our special offer.

Multi-Database Access

Oracle • SQL Server
dBASE • INGRES
Paradox
Sybase • DB2
AS/400 (SQL/400)
Btrieve • OS/2 DBM
INFORMIX
NetWare SQL
Excel • PROGRESS
SQLBase • XDB
SQL/DS
Tandem NonStop SQL
Teradata
text files
HP ALLBASE/SQL
HP IMAGE/SQL

Gateways:
IBM DDCS/2
Micro Decisionware
Sybase Net-Gateway

Need a database?

Pioneer Software's award-winning database for Visual Basic, Q+E Database/VB, is all you need to create network ready Visual Basic database applications. It's dBASE compatible too! Try it today for only **£145**

To order your copy of Q&E MultiLink VB for **only**

£325

Telephone or Fax
your order now on
Tel (0273) 483979
Fax (0273) 486224
Don't forget to add
delivery £8.00 and VAT

CONTEMPORARY Software

1 Castle Ditch Lane, Lewes, East Sussex, BN7 1YJ Telephone (0273) 483979 Fax (0273) 486224
A Division of The Contemporary Computing Company Limited

➤ CIRCLE NO. 144

32-Bit Thoroughbreds

Cliff Saran talks to six developers who aren't using compilers from the Big Three.

It is no accident that Microsoft and Borland products seem to crop up more and more on these pages. After all, they do produce the two leading C++ compilers for the PC. However, until recently, Microsoft's only offering was its 16-bit Windows 3.x Visual C++ compiler which, despite the introduction of 32-bit operating systems such as OS/2 V2.x and Windows NT, is still the 'Number One' C++ compiler for the PC. Borland's BC++ V3.1 is also a 16-bit compiler. This used to be the leading C++ development tool for Windows, but now takes second place to Microsoft. As I have said, both these products are 16-bit tools. But both Microsoft and Borland have released 32-bit C++ compilers. Microsoft's offering is Visual C++ for NT; Borland has a 32-bit OS/2 compiler.

In this article I present the views of several developers who are *not* using Microsoft or Borland compilers. Instead they have chosen the 32-bit offerings from Watcom, MetaWare and Salford.

Watcom C/C++^{32}

Watcom is renowned for producing 16-bit and 32-bit compilers which squeeze the most performance out of C and C++ code. Benchmarks against market-leaders, such as the compilers from Microsoft and Borland, usually put Watcom's offerings out in front. For years, Watcom C was considered *the* 32-bit compiler of choice for the PC. Its only real competitor in this market was Zortech C++ (now Symantec): however Watcom was always the preferred choice.

Watcom was one of the first manufacturers to release a compiler which could generate code for the Pentium. Its latest offering provides optimising technology which Watcom calls 'riscification' and instruction scheduling that improves performance on 486 and Pentium machines. It can generate code for several PC platforms including Extended DOS, OS/2 V2.x, Win32s, Windows NT, 32-bit Windows 3.x, AutoCAD ADS/ADI and Novell NLM. It can also be used, in conjunction with third party tools, as a compiler for developing embedded systems. Watcom offers three host platforms: Extended DOS, OS/2 V2/x and Windows NT.

Twice the speed...

I know from actually using both Watcom C 386 and Turbo C, that there is no contest. For me, .EXE speed is important. In one test that I performed Watcom C never performed at less than twice the speed. In some cases executables went five times as fast. I turned optimising on in both. This is no joke. Watcom C **is** *that much better. Never mind waiting for the Pentium. If you want your code to fly two to five times as fast right now, just get Watcom C.*

If I want an integrated environment, then I use Windows or DesqView. I will admit Turbo C's F-keys and on-line help is useful occasionally, but then again the environment gobbles up memory. I can use a reference book just as easily, and let's face it, for experienced C programmers, all the glitzy stuff in Turbo C is not all that useful. (I am assuming Borland C is not that different.) That 'Project' idea has to be the absolute dumbest. MAKE files are the only way to go.

As for compilation speed, of course Turbo is faster. I may have many factors, but this is irrelevant when using MAKE. On my 486/33, Watcom C will compile a 1000 line program in less than 30 seconds, which is good enough for me. My modules are usually much smaller.

Watcom's debugger may not be integrated, but it is much more powerful, providing assembly level debugging even in protected mode. The Watcom profiler is also quite nice, though it doesn't give me a pie chart like Borland's apparently does.

Watcom C comes with a large number of tools and libraries. With it you can create code for Extended DOS (royalty free extender is included), Windows, OS/2, Penpoint, AUTOCAD, and a few others. It produces very tight and fast object code, using global optimisation with a code generator that has been fine tuned for almost a decade.

Paul Hsieh '
Department of Mathematics
University of Toronto

All Intel platform

I have working with Watcom's OS/2 V2.0 C/C++ compiler for the last six months. Interleaf produces document management, desktop publishing and information management software. We are presently engaged in a project called WorldView which provides online viewing of documents. We have been porting the code to Win32 using the Windows version of the compiler. A 32-bit DOS version of our software is also planned.

Before OS/2 V2.0 we were developing for OS/2 V1.3 using the 16-bit Microsoft OS/2 compiler. The main reason for moving to Watcom was the compiler's support for all PC platforms, including 32-bit DOS, Win32 and OS/2.

We have recently upgraded the compiler to include the 16-bit DOS and Windows 3.x versions. Watcom offers the upgrade, called Delta Pak, for only $99.

It would be untrue to state that all has been well since we moved to Watcom. It was okay porting code from UNIX to 32-bit PCs, but the 16-bit code gave us some serious headaches.

Rev Up Database Programming

with the Greenleaf Database Library

NEW!

Powerful Functions

- ❑ The *SoftC* Database Library is now the new Greenleaf Database Library.
- ❑ Unsurpassed speed and flexibility for access to industry-standard database data, index and memo files at an affordable price.
- ❑ Databased Advisor says, *"SoftC Database Library has top ratings!"*
- ❑ Don't be fooled by pretty ads: the Database Library is all you need to interface with dBASE III, dBASE IV, FoxPro, FoxBASE, Clipper, dBXL, and other xBASE files including new FoxPro CDX index files.
- ❑ Supports MS-DOS and Windows, and is portable to OS/2 and UNIX.
- ❑ Includes Windows 3.1 DLL, supports Microsoft, Borland and Zortech C/C++.
- ❑ Single and multiple user; network access fully supported.
- ❑ Database package not required: this product is a complete ISAM library.
- ❑ Windows DLL linkable with most DLL-capable compilers (including Visual Basic).

All Greenleaf libraries feature:

- ❑ No royalties
- ❑ 90-day money-back guarantee
- ❑ FREE unshrouded source (ANSI & K&R)
- ❑ Top rated documentation AND online documentation with FREE help engine!
- ❑ FREE BBS access
- ❑ FREE unlimited UK technical support

Database Library version 3.22 £249 + VAT

☎ Telephone today for complete information, demonstration, or details of how to order.

0566 86037

FAX 0566 86147

Citadel Software Ltd

Coombe, Trewen, Launceston, Cornwall PL15 8QF

GREENLEAF Software©

➤ CIRCLE NO. 184

The Ultimate in Communications Programming

NEW: Comm++ 2.0

- ❑ C++ class library for MS-DOS, Windows 3.1 and OS/2.
- ❑ Device independence through inheritance.
- ❑ Supports Microsoft C/C++, Visual C/C++, Borland C/C++ and other popular compilers.
- ❑ COM1 ... COM8, standard and intelligent multiport boards.
- ❑ Baud rates to 115,200; drives 8250, 16450 and 16550 uarts including FIFO controls.
- ❑ XON/XOFF, RTS/CTS and DSR/DTR handshaking.
- ❑ XMODEM, YMODEM, ZMODEM, Kermit and ASCII file transfer protocols; 1K, R, RLE compression, crash recovery, and more.
- ❑ VT52, VT100, ANSI and TTY terminal emulation.
- ❑ Hayes modem control classes.
- ❑ Screen drivers include TextWindows, DataWindows, Zortech and compiler independent functions.

PowerComm

- ❑ PowerComm Toolkit for Windows v1.0: CommLib 4.0 DLL and VxD.
- ❑ Drives most standard multiport boards; share board among Windows 3.1 and/or DOS apps.
- ❑ Throughput multiplier: much faster than Windows or other drivers. The VxD is written using CommLib's fast assembler code in 32-bit flat model. **Requires CommLib 4.0**

Greenleaf CommLib v4.0 £249 + VAT
Greenleaf PowerComm v1.0 £139 + VAT
Greenleaf Comm++ v2.0 £199 + VAT
CommLib Professional (CommLib with PowerComm) £379 + VAT

☎ Telephone today for complete information, demonstration, or details of how to order.

0566 86037

FAX 0566 86147

Citadel Software Ltd

Coombe, Trewen, Launceston, Cornwall PL15 8QF

CommLib 4.0

- ❑ Level 2 device independent C functions for these drivers: MS Windows, Greenleaf Standard, Fast, Smart, DigiBoard, Smart Arnet, Smart Star Gate, Sparkle, BIOS, Extended BIOS, Polled, MODEM Assist, FOSSIL.
- ❑ Windows 3.1 support includes TextWindows: easy programming.
- ❑ 286 Protected Mode for Phar Lap and Rational Systems DOS extenders.
- ❑ XMODEM, YMODEM, ZMODEM, Kermit, ASCII protocols: many options.
- ❑ Unlimited number of ports for ISA, EISA and MCA.
- ❑ Modem controls, keyboard, screen, handshaking and much more.

GREENLEAF Software©

➤ CIRCLE NO. 145

I don't use the Watcom debugger: Turbo Debugger is much better. As for an IDE. I prefer makefiles myself, but last year I saw a demo of Watcom's new GUI-based IDE, which looked very impressive.

Watcom's technical support isn't bad. Although we usually end up having to leave a message, someone from Watcom always calls back within a day or so. We also like Watcom's BBS. For instance I used it a few days ago to download a patch for the V9.5 C/C++^{32} compiler. Watcom can also be contacted through the Internet.

Jonathan Arnold
Interleaf
Massachusetts

MetaWare High C/C++

Unlike Watcom, MetaWare compilers offer more than multi-platform support for Intel 80x86 architectures. They are available on Sun Sparc systems, UNIX SVR4, Intel i860, and IBM AIX/ESA and HP 9000 Series 300/400 and 700/800.

One unique feature of MetaWare's High C/C++ compiler for DOS/Windows is its specific support for Weitek Abacus maths co-processors such as the 1167, 3167 and 4167 as well as 387-compatible chips. The Weitek floating point co-processors give up to three times the performance of Intel's 80x87 family.

MetaWare provides eight levels of global such as loop unrolling, subexpression elimination, spill analysis, low-level instruction scheduling and the ability to pre-load arguments from memory. The compiler features ANSI conformance, Lint-like checking of C source and objects which are binary compatible with IBM's SOM (System Object Model). In addition, High C includes the Rogue Wave Tools.h++ class library and the MetaWare Application Development Kit (ADK) for Windows 3.x. Tools.h++ includes classes for linked lists, B-tree disk retrieval, string classes and other templates. The ADK enables developers to write 32-bit applications on Windows 3.x using a special 'translator' called a *supervisor*.

Weitek co-pro support

Micropath writes software which enables telephone engineers and consultants to simulate and analyse radio link systems such as mobile radio or microwave links. We began in 1985 and moved to MetaWare in 1988. At that time the MetaWare C compiler was the only 32-bit DOS compiler available.

We have found that MetaWare High C/C++ generates particularly good code, especially in areas where optimisation is critical. Moreover, it is the only compiler which supports the Weitek 4157 maths coprocessor. We rely heavily on numerical simulation. Thanks to its ability to generate Weitek floating point instructions, we have found that code compiled with High C/C++ performs extremely well.

Although we generate all libraries inhouse, we use the Vermont View GUI library to build the user interface. In this respect, we haven't encountered any difficulties integrating Views with the MetaWare code.

While MetaWare offers both a C and C++ compiler, at Micropath we have generally stuck with C, as a C++ standard does not yet exist. However, we haven't discarded the MetaWare C++ compiler. While evaluating it, we discovered that its overload operator facility worked with global `new` *and* `delete`. *Put simply, this means it is possible for a developer to 'plug-in' an alternative heap manager in MetaWare C++. As far as I am aware, the USL CFront compiler V3.0, is the only other C++ compiler of which I am aware, that supports this feature.*

Despite developing only on 486s, we find it reassuring to know that High C/C++ is available on several platforms. We are also impressed by the people on the MetaWare technical support team. If they can't solve your problem immediately, someone will always get back to you.

Christopher Batory
MicroPath
Quebec

Lint-like diagnostics

We chose the Metaware High C/C++ compiler, after evaluating it against Zortech and Watcom compilers. As a rough guide to performance, Watcom, which has a reputation for generating fast executables, came out 32% slower than High C/C++. Sairen Technologies develops CAD/CAM software, where execution speed is of paramount importance. Our 3D graphics code is highly optimised, so we need a compiler which will give us the fastest code possible.

Our software is developed entirely in C++ using templates and multiple inheritance. The MetaWare compiler doesn't yet support exception handling: IBM C Set/2 and Watcom V/C++^{32} V9.5 are the only two PC compilers I have come across that support this feature.

We develop under both DOS and Windows. With DOS we use Meta Graphics and C-Scape libraries. zAPP (from Inmark) is used for developing Windows GUI. Although Inmark doesn't support MetaWare directly, we found that it only took a day of juggling to get zAPP to work with High C/C++. Not bad considering these things can sometimes drag on for weeks.

The diagnostic messages which accompany compilation errors are excellent. Not only does MetaWare point to the line on which the error occurs, it also gives the column number too. It even suggests a possible reason for the error such as 'missing semicolon'. In line with true ANSI C compliance, High C/C++ provides true non-type casting. For instance a function `fn` *declared as:*

```
void fn(char ch, int i);
```

can be invoked:

```
fn(char, char);
```

according to ANSI. The compiler is supposed to resolve the type-cast automatically.

There are two debuggers shipped with High C/C++: a source-level interactive debugger and a remote debugger. Since we spend most of our time developing graphics applications, the remote debugger is our preferred choice since it eliminates the need for a second monitor being attached to the development PC.

Although MetaWare is based in the States, we have not encountered any problems getting technical support. MetaWare also offers a BBS and can be accessed from Internet.

Sairen Technologies was established in 1989. It develops custom CAD/CAM software for several large customers including Rolls Royce and Lucas.

Richard Guelbert
Sairen Technologies
Nottingham

Is This What You're Looking For In a CASE Tool?

- SSADM v3 and v4 Support
- Object-Oriented Support
- PowerBuilder Interface
- Visual Basic Interface
- Gupta SQL-Server Interface
- Schema Generation
- Reverse Data Engineering
- Documentation Facility

...and more!

To find out how much more, for how little phone now

(0926) 450858

SYSTEM ARCHITECT™

The leading Price/Performance CASE tool.

PC-based, feature-rich, easy-to-use, powerful and affordable.

POPKIN
SOFTWARE & SYSTEMS
INCORPORATED

Real Techniques and Methods Ltd
118-120 Warwick Street Leamington Spa
Warwickshire CV32 4QY

MICROSOFT™
WINDOWS™

➤ CIRCLE NO. 146

Salford C++

Many of you will probably already be familiar with Salford's stable of 32-bit Fortran compilers. The company recently introduced a 32-bit C/C++ compiler which claims to offer fast compilation speeds (up to 100,000 lines per minute on a 486 DX2, according to Salford).

Unlike MetaWare or Watcom, Salford developed its own 32-bit DOS extender called DBOS which features a 2 GB address space, dynamic link libraries for DOS and a disk cache mechanism which Salford claims will improve overall system performance and optimise the use of memory.

Salford C++ provides a number of run-time error checking features to cut down common programming bugs. The `NULL` pointer is represented in such a way that it causes a run-time error if an executing program attempts to reference it. In addition it provides a mechanism for detecting references through unset or dangling pointers.

Using a preset bit pattern for unset variables, the Salford C++ debugger can also pinpoint when a variable has been used before it has been assigned.

Since Salford C++ is a 32-bit compiler, it cannot directly access 16-bit Windows 3.x code, so Salford provides a library called ClearWin which enables developers to write 32-bit applications. These can call the 16-bit Windows 3.x SDK. It offers three levels of complexity. First there is simple screen output. Statements such as `printf()` are routed to Windows output functions. ClearWin manages Windows automatically, including features such as scroll bars, without the need to modify source code. At the second level, it provides developers with a set of window management routines for sizing and positioning. Developers can add menus and graphics without having to undertake full Windows development. Finally ClearWin provides access to the Windows API.

NULL pointers...

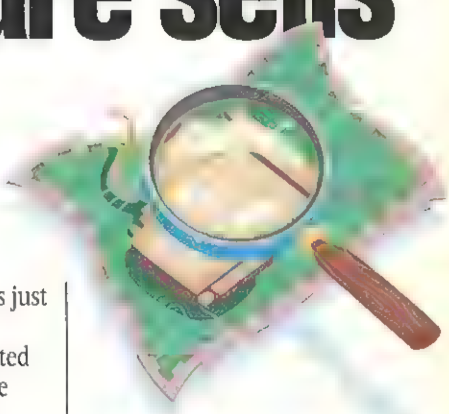
Despite the proliferation of C and C++, Fortran remains very popular in engineering fields. We have been using Salford Fortran at Vector Fields for many years, producing multi-platform graphics libraries for UNIX, PC and VAX VMS. Since the launch of the Salford C++ compiler we have been running C and Fortran development in parallel.

Salford C++ offers the ability to check source code to a high degree. It provides a way to trap null-pointer assignments at run-time.

There is a comprehensive SVGA graphics library and ClearWin, a library which enables DOS programs to be run as Windows programs with no alteration to source code. Salford C++ is shipped with a 32-bit DOS extender called DBOF which offers virtual memory management. In addition, it supports dynamically loadable libraries for DOS. These are similar in function to Windows DLLs, but don't need an error-prone EXPORT file. I have found this feature very easy to use: a simple compiler switch reduces the overall .EXE size significantly.

Compilation and link speed of Salford C++ are faster than any other compiler I have used. In terms of performance,

Why some software sells more than others.



Success. All software developers strive for it. Now, Don Gall was on top of the world. Software protection made all the difference. Especially in Europe and Asia. Sales were four times better than before. He is the founding father of Sentinel — the guru of software success.

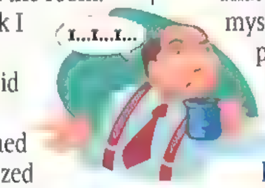
Struggling Software Sales

One day, trekking through the coffee fields of Java, Don ran into his old college buddy Simon Seagull. "Don, my sales are well below expectations," Simon explained his plight, "My software should sell like yours, Don!" Yet despite critical acclaim Simon's company, SimonSays Software, teetered on a financial tightrope. "What's your secret, Don?"

They spent hours analyzing potential problems. They looked at everything.

The Key to the Problem

Finally, Don leaned back and asked the assumptive question, "What about protection – are you using Sentinel?"



Nervously, Simon sipped his coffee. His hands shaking as his eyes darted the room. "No. I didn't think I needed to."

Don's chair slid out from under him and he crashed to the floor. Amazed in disbelief, Don cried, "You What?!" Grabbing his tattered scrapbook, Don pulled out photos of his travels. "Ever been to Seoul? Prague? Anywhere? Ten bucks will buy you anything, even bootlegged copies of software."

Don's Road to Success

Thumbing through the scrapbook, Don shared his experiences. "Back in the '80s, I was in your shoes – beaten, battered and bruised." Simon listened. "Then, after a heart breaking trip around the world, I called the Software Publishers Association (SPA)."

"I could hardly believe it. They told me developers lose billions of dollars each year. Why? Illegally copied software. In some countries there are nine pirated copies for each legal copy sold."

Simon was disgusted, "It's just not fair."

"That's why I committed myself to solving the piracy problem," explained Don. Simon's eyes lit up. "The dongle!" he shouted. Don corrected him, "Not just any dongle — the dongle that paved the road to success for over 10,000 developers worldwide — Sentinel."

Successful Developers Use Sentinel

Don pulled a stack of letters out of his gunny sack. "All of these people tell the same story." Don read about a successful developer from California who swears she wouldn't be in business without Sentinel. Another company says protection costs less than litigation, plus they don't have to spend time and money supporting illegal users.

Others confessed they wouldn't market products internationally without protection.

The hours flew by, story after story, Simon learned Don Gall's secret. To succeed is to protect. To protect is to secure with Sentinel.

Most Advanced and Widely Used Dongles in the World

Backed by the world leader in software protection, Rainbow Technologies, the Sentinel Family of hardware keys is the most diverse and comprehensive selection available. For DOS, Windows, OS/2, Macintosh, LAN, UNIX and others. They're simple to install, and are the most reliable and compatible available.

Rainbow offers just-in-time delivery and the largest technical support and engineering staff in the software protection industry.

Call Don Gall today for a free copy of **"The Sentinel Guide to Securing Software."** Or better yet, ask him for a low cost Sentinel Evaluation Kit — complete with a working dongle!

For your free guide to securing software, call your local office or distributor listed below.

SENTINEL
Securing the future of software

When you need a dongle, you need Sentinel. The only dongle Don Gall will use.

RAINBOW
TECHNOLOGIES

RAINBOW LTD./U.K. 44 932 570066 ▪ **RAINBOW MICROPHAR/FRANCE** 33 1 47 38 21 21 ▪ **RAINBOW GmbH/GERMANY** 49 89 961 3051 ▪ **PACIFIC RIM/USA** 714/454-2100

Worldwide distributors: ARGENTINA: Agri-Aid, S.A. 54 1 46 8364 • AUSTRALIA: Fagan Microprocessor Systems 61 3 699 9899 • BENELUX: Introcom Security Key 31 74 430 105
BRAZIL: Mips Sistemas Ltda. 55 21 507 1839 • BULGARIA: KSIMETRO 3592 79 14 78 • CHILE: ChileSoft Ltda. 56 2 639 8892 • COLOMBIA: Construdata 57 1 610 8549
CZECHOSLAVAKIA: ASKON International 42 2 36 44 25 • GREECE: Byte Computer S.A. 30 1 923 2335 • HONG KONG: Beta Trading C & P 852 515 0018
HUNGARY: Polyware Kft 36 76 481 236 • IRAN: Gam Electronics 982 122 2374 • ITALY: BFI IBEXSA SPA 39 23 31 00535 • ITALY: Siosistemi 39 30 24 21074
JAPAN: Giken Shoji Co., Ltd. 81 52 972 6544 • JORDAN: CDG 962 6 818360 • KOREA: Genesis Technologies Corporation 82 2 578 3528
MEXICO: Impex Computacion, S.A. de C.V. 52 66 210 291 • NORWAY: Perico A/S 47 2 49 1500 • PERU: Computadores Andinos, S.A. 51 14 42 4888 • POLAND: DAGMA SP.ZO.O 48 32 525 200
PORTUGAL: HCR 35 1 135 61865 • SAUDI ARABIA: ZEDAN 966 2665 1904 • SPAIN: MECCO 34 3 422 7700 • SWITZERLAND: IBV AG 41 1 741 2140 • SWITZERLAND: Safe Compaid S.A. 41 2 421 0385
THAILAND: BCS International 66 2 319 4451 • TUNISIE: ICS 216 1 334 417 • TURKEY: BIMEKS, Ltd. 901 348 3508 • VENEZUELA: H&R&T Osers 58 2 261 4282

➤ CIRCLE NO. 149

source compiled with Salford C++ gives comparable performance to that compiled with the NDP 32-bit C++ compiler.

Adam Dawson
BUSS Ltd
Bradford

Multi-language support

GKS writes 2D and 3D portable graphics libraries such as GKS and Phigs for hardware ranging from PCs to Crays. The big machines usually have their own compilers, but on the PC we chose the Salford development tools.

Salford's Fortran compiler has a reputation that's second to none. It's extremely fast.

About a year ago we introduced Salford's C compiler to compliment are existing Salford Fortran development tools. This was because we needed some way for our Fortran code to take full advances in the operating systems of today. How often do you see an operating system written in Fortran? But C is everywhere.

So when looking for a compiler, our primary concern was for mixed-language support, especially between Fortran and C. We have found that Salford's developments tools work extremely well together. GKS did consider Watcom. However, in our opinion, the Watcom Fortran compiler isn't very good.

The C compiler itself supports both the K&R and Standard C calling conventions. It is shipped with a good Windows compiler and debugger. There's also a royalty-free 32-bit DOS extender included.

We have found Salford to be a very understanding UK-based compiler vendor. Technical support is excellent: we usually get an immediate reply. Unlike companies based in the States, we have encountered no problems getting in touch with Salford. There is also a BBS.

David Findlay
Scientific Software Ltd

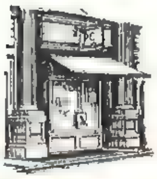
Conclusion

While companies such as Microsoft and Borland concentrate on the mass market appeal of pretty Windows-hosted GUIs and integrated development tools, the compilers in this article offer only command-line interfaces and the ubiquitous MAKEFILE.

Not that lack of an IDE makes much difference. The people who use these compilers aren't looking for pretty GUIs: they expect no-frills, tip-top performance.

Although I have concentrated on only three compilers in this article, there are, of course, several other 32-bit C++ compiler manufacturers. But, in general, these tend to be less widely used. Among them are: TopSpeed (Clarion) C++ and Glockenspiel C++ (Computer Associates).

Watcom C+/C++^{32} V9.5 costs £375. Lattice High C/++ 386 V3.1 is priced at £555. Salford C/C++ costs £695.



Hardlock E-Y-E® – tying the hands of software pirates

The effective way to protect your software

FAST Electronic has made life a lot harder for software pirates. Hardlock E-Y-E was designed using cryptographic principles. It took the experience and know-how of Germany's number one in software protection and the leading edge technology of a US semiconductor company to create the ultimate software protection tool.

The technology programmers have at their fingertips

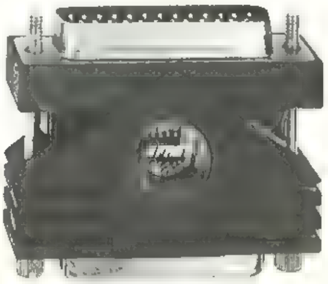
Hardlock E-Y-E is based on a custom chip and combines all the features that a programmer would expect from such a device: secure, algorithmic query routines and an optional non-volatile memory for custom configurations. With the Crypto-Programmer card from FAST you can program the algorithmic parameters and the memory within seconds. This unique card guarantees that no one else can burn Hardlock E-Y-Es with your codes. Linking Hardlock E-Y-E to your software is easy: you can either protect your .COM and .EXE files with the automatic encryption software HL-Crypt or integrate FAST's high level language routines into your source code.

Readily acceptable to your customers

Hardlock E-Y-E allows unlimited backup copies of the master floppy. The customer gets the device together with the software and plugs it into the parallel port between the printer and the PC. Daisy-chainability, outstanding reliability and the compact high tech design guarantee that your customers will accept Hardlock E-Y-E.

The benefits your management will appreciate

Hardlock E-Y-E can be programmed by the software house with the Crypto-Programmer card. This ensures optimum delivery schedules and stock flexibility. Revenues will go up as software piracy and multiple usage are prevented.



**Hardlock E-Y-E
programmable, algorithmic response
and memory option – all in one**

MAGNIFEYE

**Order your demo unit today. Contact Magnifeye,
235-239 Walmer Road, Walmer Studio # 6, W 11 4 EY,
Telephone 071 221 8024, Fax 071 792 3449.**

Magnifeye is a subsiduary of Fast Electronic GmbH.

➤ CIRCLE NO. 152

Don't forget your ID

Mike Lee discusses the problems with the ever-increasing number of IDs which users are forced to remember.

The concept that the User ID identifies the user is not only false but is now standing in the way of progress. Electronic mail packages and other digital communications methods mean that User IDs have to be public knowledge. If we are to contact someone via such media, the equivalent of their telephone number or address needs to be readily available. Consequently, the problem has evolved into a major barrier.

New office locations, company acquisitions and mergers mean that the problem is compounded further by mixing standards and the effects of duplication. Can anyone still identify the true identity of the user from the User ID?

Strangely, the main factor that establishes, declares and controls our interface with a Data Processing facility - the User ID - has not historically been a prerequisite to using computers. Originally, computers were used by operators who ran batch processing. Batch Jobs were identified by the Jobname, a label which lasted for the duration of the job. Some operating systems were able to keep records of the jobs that ran and were even sophisticated enough to prevent duplicate job IDs; ironically some current computer systems do not have the capability to prevent duplicate User IDs being logged on at the same time.

Very few PCs or operating systems in use today ask the person who is attempting to use it to identify themselves. The assumption is that the act of powering on the PC amounts to authorisation to use it. So it is often left to the application to prevent misuse by asking for a User ID and an associated password.

There is, however, a known and widespread problem with all User IDs. They have to be meaningful and reflect the identity of the user who uses it, his location and if possible his type of work. However they also have to be memorable in order to simplify the process. UserIDs do not have to be a secret code.

UserIDs do not have to be a secret code security is the job of the password

If the convention chosen is working correctly, the probable User ID of any user in the company should be available by isolating a few pertinent details about the individual concerned.

Some concerned users believe that this can cause problems with illegal access, but fail to appreciate that security is the job of the password. When used correctly in conjunction with the User ID, the password should ensure reasonable security for most jobs of work on the computer. The only problem that deducing a User ID causes is when someone attempts to enter a computer system with a guessed or deduced User ID and is stopped by a password, but continues with the attempt until the User ID in question is locked or revoked. If the ID belongs to an administrator, extreme cases may result in loss of service until a Master UserID can be used.

When UserIDs are designed there is a temptation to tailor them to the application in use rather than to the person who is going to use them. For example, the User ID `PAYFB01` for Fred Bloggs accessing Payroll is fine as long as there are a limited number of users on a small number of applications. However, if Order Entry was implemented, the tendency would be to set up another User ID convention. The problems are obvious - staff who have dual functions end up having two or more User IDs, and therefore passwords, to keep safe.

Such are the risks taken by failing to conform to a naming convention. One large company holds a record of a particular user who has 26 User IDs assigned to his use. He keeps all the UserIDs, the systems they access and the associated passwords written on a whiteboard near his desk. The difficulty in resolving a problem such as this lies in introducing a secure, productive naming convention and changing all of the applications involved. This would be a mammoth task.

As more operating systems and hardware platforms are introduced into the business environment, either due to expansion or intolerance with existing platforms, the more confused the User ID issue becomes.

For example, to connect a PC via a LAN to a mainframe is a common event. The most popular LAN in use today has a three character User ID only. This immediately causes problems with naming conventions and in most installations requires the user to remember two disparate User IDs and, of course, associated passwords.

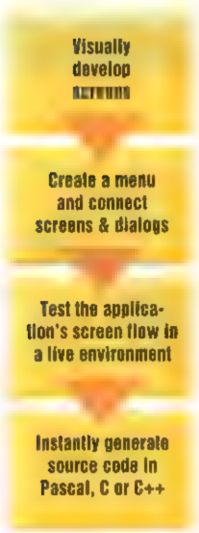
There is no agreed standard, nor is there likely to be, concerning the use or construction of User IDs The more hardware platforms that are in use, the more User IDs a user is required to have. This causes frustration on the

Prepare Yourself for Visual Reality.

Introducing ProtoGen+. Visual tools with the most awesome workbench ever created for Windows development.

he future has arrived—a complete point-and-click, WYSIWYG workbench that lets you create dazzling applications without writing a line of code.

Discover the ease and productivity of visual development!



Visually develop [illegible]

Create a menu and connect screens & dialogs

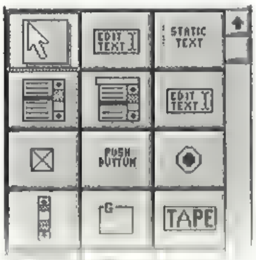
Test the application's screen flow in a live environment

Instantly generate source code in Pascal, C or C++

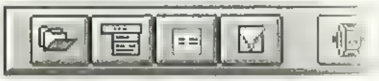
Paint your screens. Design a menu and link screens together. Test the flow in a live environment, and generate code for ANSI C, C++ for OWL or MFC or Pascal with objects. It's that easy.

And this open! ProtoGen+ will work with your database, compiler, libraries and extensions. Powerful add-on features, like SQLView offer workbench access to multiple databases to develop client/server and xBase applications. Snap-in components make ProtoGen+ open to future development technologies—whatever they may be.

ProtoGen+ is easy to learn, speeds your development cycle and protects your investment by generating C,C++ and Pascal. We guarantee you'll prototype and generate exciting applications faster than you ever thought possible!

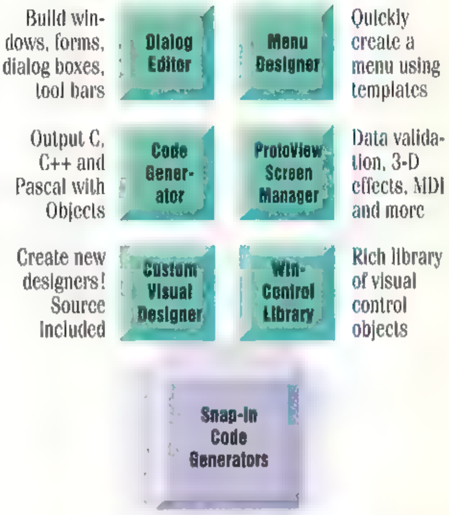


Point-and-click to paint screens with bitmaps, icons, tables, data validation, custom colors, fonts, 3D effects, visual toolbars, status lines, balloon help, MDI and more!



Bring your applications to life using the latest visual design tools!

The most powerful, open set of Visual Development Tools ever!



Build windows, forms, dialog boxes, tool bars — Dialog Editor

Menu Designer — Quickly create a menu using templates

Output C, C++ and Pascal with Objects — Code Generator

ProtoView Screen Manager — Data validation, 3-D effects, MDI and more

Create new designers! Source Included — Custom Visual Designer

Win-Control Library — Rich library of visual control objects

Snap-In Code Generators

License our technology to create new code generators

30 DAY

Fasten your seatbelt for ProtoGen+!

only £149 (list price £299)

Tel: 0273 483979

Fax: 0273 486224

ProtoGen+'s SQLView access to multiple databases is available at an additional price; ask about it when you call.

CONTEMPORARY *Software*

1 CASTLE DITCH LANE, LEWES, EAST SUSSEX, BN7 1JY
TEL: 0273 483979 FAX: 0273 486224

➤ CIRCLE NO. 153

part of the user because the User IDs he uses are invariably written down on the terminal he uses for quick reference. This appears to be a normal and accepted practice.

Remote siting of users and centralisation of processing adds further problems regarding the identification of users. It is desirable that a remote User ID gives some kind of indication of the location, especially when logging and tracking is used within an application. Also more and more security and access control products restrict access times.

Therefore, it can be strongly argued that with most of the User ID naming conventions in use, the service provider has to use his own, sometimes detailed, knowledge of the user base or a translate table of some kind to identify the user, the location, the outline security profile and the priority or urgency of the situation.

If a user has two or more User IDs then it is likely that for the majority of the time, he will have two or more passwords. It is consequently more difficult to enforce any procedures designed to keep passwords discreet and secret. If the user has some five or six User IDs then, unless the passwords are identical, he will need to record them somewhere as they will be difficult to remember.

The need to rationalise User IDs to conform to a naming convention that is already in existence in the company is strong. Many companies identify either the staff or payroll number as the ideal identifier to use. Adding a prefix or a suffix with a location digit and, if appropriate, a four digit telephone extension number, gives the ideal method of identifying the person, his location and the contact number in the event of needing to contact him.

Depending upon the installation, there are a number of problems with adopting this convention. The length of the User ID is the most obvious, including all aspects of the convention could result in a User ID stretching to a dozen digits. The second, and probably most costly, problem is how easy it is to convert all of the User IDs that are required by the applications to the same naming standard.

There are three options open to the service provider that will alleviate the immediate problems. They could do nothing and live with the problem which is what happens in most situations. In the event of using multiple User IDs, the passwords can be synchronised automatically so that the same password is used for each User ID wherever it resides and for whatever it is used. Finally, a new User ID and discreet password could be introduced. The existing User IDs could then be driven into the background using a control facility or session manager. This will either log on to the application using the existing password for the old User ID (which will necessitate 'freezing' it) or will require synchronisation with the new password.

.EXE

Mike Lee has 26 years experience in the IT Industry and currently works as a Product Manager for CKS at its Bracknell office, specialising in access control and security solutions. He can be contacted on 0344 868 868.

Does your left hand know what the right is doing?

Get your hands on Intasoft's Configuration Management tools and it will!

SMS is our full-featured version control package: perfect for preventing users overwriting each others' changes and for automating the build process – as well as telling you *Who?*, *What?*, *When?* and *Why?*

Full configuration and change management can make anyone all fingers and thumbs. Not with **AllChange**, Intasoft's brilliant package which provides facilities for product design, baselines, life-cycles, user roles, change requests, configuration build and release management.

Our products will help you with standards (e.g. ISO9001, BS5750) and are backed by our outstanding support.

With all of this available for MS-DOS, Windows and UNIX at prices from as little as £490, we think you'll agree Intasoft are handy people to know!

INTASOFT, Tresco House, 153 Sweetbrier Lane, Exeter, EX1 3DG.
Tel 0392 217670 Fax 0392 437877

INTASOFT

Quality tools for professional software developers

NEW *SMS Version 4 for Windows Now available*

NEW *AllChange – Change control & configuration management system*

➤ CIRCLE NO. 155

Window Shopping

It's not often that Jules telephones anybody. Here, he explains why not.

Did you know that there are such things as shop-window competitions? Sometimes they're organised by towns, other times they're organised by trading associations, but the idea is always the same. A group of shops is judged and the one with the best window display is declared the winner. Window displays are part of advertising. They are as much a part of the shop's image as the sign over the door, the logo on the letterhead and the smile of the assistant inside. Why competitions are necessary is beyond me, because a good window display is one which brings in the customers. It doesn't take a great statistician to recognise when a display is having a positive or a negative effect on sales.

Some companies, however, don't have a shop window at all. There are several reasons for this; either the company itself is too diffuse to have a site which can be dressed or the products being shown are simply not sexy enough or the company doesn't want its customers dropping in for fear of what they might find. Such companies still have to provide an image to their customers; their shop window is, in practice, their telephone switchboard.

Few companies realise this. Their front-line telephonists are badly-trained, don't understand the company structure and don't understand the product line. They're surly and short. The larger the company, the worse this becomes.

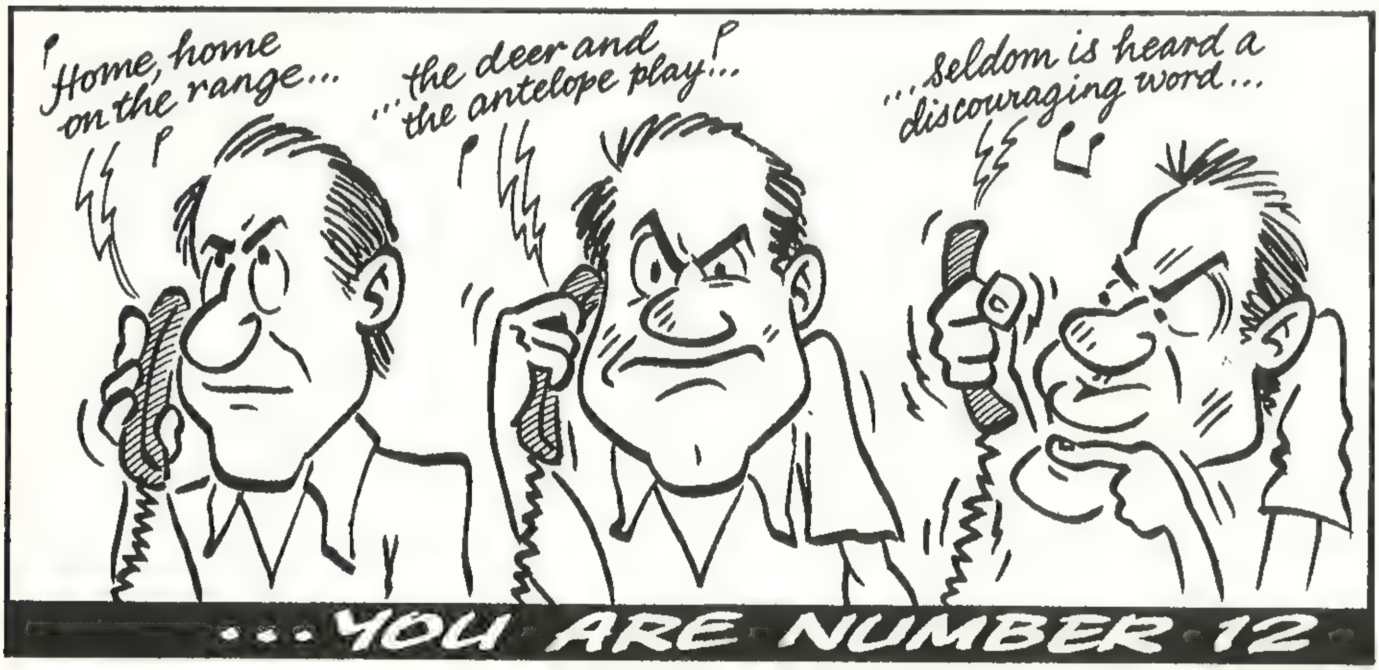
'I'd like to speak to tech support, please.' What is hard about that?

After a pause: *'I'm sorry, he's being told off for not answering his phone. Can I put you on hold?'*

Ah, the phrase that excuses any sin, and any rudeness; *'Can I put you on hold?'*. It's not a question, it's an injunction. It means I'm not interested in helping you to find a solution, I'm not interested in finding someone who can help you, I'm not going to speak to you, so don't even try to speak to me. I'm not interested in you. Then, you'll be subjected to the Rolf Harris Stylophone version of *Home on the Range*, or the Patagonian Nose Flute Ensemble version of *Pop Muzic*, or something equally inappropriate.

When you are about to connect the phone across the mains, desperately trying to make it stop, the operator comes back, with a worse injunction: *'Still no answer, will you continue to h...<click>'*, and the damned music starts again. Of course I'll continue to hold. If I don't, I'll have to call back and suffer another twenty minutes of this. It's happened!

Then the phrase I've been waiting for, the line without which my life would be unfulfilled: *'Putting you through now.'*


Home, home on the range...
...the deer and the antelope play!...
...seldom is heard a discouraging word...
...YOU ARE NUMBER 12

And after a pause, *'Hello, Boiler room.'*

'What? No, I want tech support.'

I then listen intently to a distant conversation on the other end of the line to which I am not party, although someone appears to be asking: *'Anyone here heard of a guy called Support? Yes, it is a stupid bloody name, but he's wanted... You want me to ask him if he's a crank?'*

With still no chance of my saying anything, he eventually comes back on the line, with the words: *'Never heard of him. I'll put you back to reception. I'll have to put you on h...'*

And more *Home on the Bloody Range*, interrupted only by the words:

'Hello, kitchen...'

And there I am, blown on the winds of blind fate around this telephone system like the Flying Dutchman, occasionally sighted in the Department of Invoice Docketing or the Department of Disinformation or the Department of Losing the Setup disk for the Upgrade Offer, but always banished with the words *'Putting you on h...,'* cast adrift again with no hope of making port.

You idiots: do you think I've got nothing better to do than listen to *Home on the Range* on the phone? In the length of time it takes for you to answer your phone, I could have bought the record or even bought a stylophone and learned to play it myself. Does it not occur to you that I might want to make a cup of tea or have a leak or take a correspondence course in the length of time it takes to get through to someone?

For *any* company to do this is reprehensible. For a computer company, which is supposed to know about machines, it is unforgivable. It can't be beyond the wit of man to invent a machine that comes on the line and says: *'There are* five *people and a beetle ahead of you in the queue. From recent experience, you will probably have to wait* four days *for a reply. Please continue to h...'* Queues in telephones are not like real queues, you can't wander off for some mood-elevating chocolate and come back again. You can't have a conversation with the people beside you.

Ah, there's an idea. How would it be if, instead of putting you on hold, you were connected to a chatline group? 90% of all problems would be solved collaboratively before an employee had even picked up the phone. And if someone was placed into the wrong queue, it would be apparent almost immediately and he could call for the operator to put him somewhere more sensible.

'Hello, can I speak to tech support, please?'

'No, I'm afraid not, but I can connect you with thirty other people who also want to speak to tech support. Is that acceptab...'

'It's not acceptable, but it's a start.'

'Oh, look guys! A new visitor! Get ready for a long wait - I've been here nearly three years.'

'Good grief, three years for tech support?

'Tech support? This is the queue for the upgrade offer.'

'No, it's the invoicing department.'

'No, it's the toilet.'

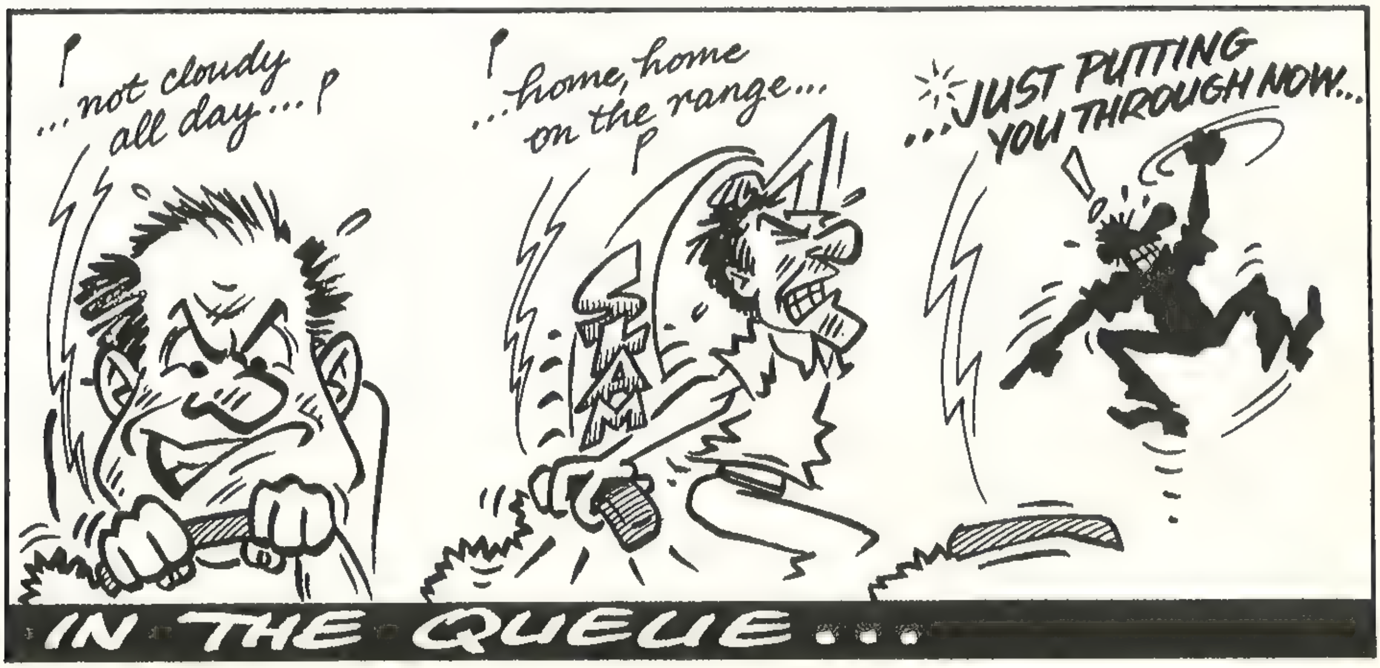
Actually, it's none of those, it's the oubliette.

'Oi, what are you lot doing in my kitchen?'

I guess we're all wrong. No, on reflection, chatlines wouldn't work either. It's bad enough for its customers thinking the company is full of gits. Telling each other they thought so would be quite beyond the pale. Perhaps it would be easier to make the product work in the first place.

EXE

Jules has a pleasant telephone manner, but an utterly intransigent answering machine, which you can hear for yourself by calling 0707 644185. It would probably be easier to mail him on CIX as jules, though.


...not cloudy all day...
...home, home on the range...
...JUST PUTTING YOU THROUGH NOW...
IN THE QUEUE...

Polymorphic Persistence in C++

Stick around while Laine shows you how to make your objects stick around...

Although C++ has all the whistles and horns necessary for dealing with objects in memory, it is completely lacking in ability to store those objects on disk so that they can survive between runs of a program. This ability is called 'persistence', and it is very often desirable. Anything from a program's setup data to its menus and dialogs, even a smallish database, can be handled quite well with persistent objects. Unfortunately, C++ doesn't have them.

Ah, but if Bjarne had meant us to program only using built-in features of C++, he would have disallowed class definitions and included MFC V2.0 in the definition of the language! It can be quite simple to add persistence to objects in C++, as we will see in the following explanation.

Keep in mind that I am presenting an extremely simplified version of persistent classes. Although `tPersistent` can handle *polymorphic* reads (don't worry, I'll explain what that means in a minute) and collections of objects, it isn't capable of some of the more advanced features found in Turbo Vision's `TStreamable` classes, MFC's `CObjects`, or Rogue Wave Tools.h++'s `RW` classes. These packages take care of things such as multiple references to the same object, as well as using overloaded operators to implement the stream I/O operators << and >>.

Even those libraries are no comparison to a true object-oriented database, such as POET (see July and August 1992 *.EXE* for information on a previous version of this package), which even take care of such problems as object identity, simultaneous access, automatic reading and writing of referenced objects, and various indexing and iteration methods.

But sometimes you don't need that much power. You can get by with something less complex, easier to maintain and *cheaper*. Now, don't get me wrong: my example persistent classes aren't even in the same league with the above mentioned professional packages! Still, everything has its place.

Also, don't be confused by the term *persistent*. A persistent object does not automatically store itself on disk when you delete it (or when you exit the program). Likewise, it does not magically appear in memory the next time you run the same program. You still must consciously make the decision that a particular object should 'stick around', by calling its `Write()` function; later you have to call its `Read()` function to get it back again. The purpose of persistent objects is not solely to automate storage and retrieval of objects, but to make this storage and retrieval easier and better integrated with the concepts of OOP.

Monomorphism?

At the simplest level, you could add a virtual `read()` and `write()` function to each class definition. `read()` would first read in the data items defined in the current class, then make a call to the `read()` function of the base class. The result will be that objects of derived classes will be able to write themselves to disk without needing to know detailed information about their base classes' data. If you

```
#include  <stdio.h>
#include  "tregistr.h"

const int _TPERSISTENT = 1;

class tPersistent
  {
protected:
  virtual int __read(FILE *fp)
    { return 1;}
  virtual int __write(FILE *fp)
    { return 1;}
public:
  static tPersistent *build()
    { return new tPersistent; }
  virtual int ID()
    { return _TPERSISTENT; }
  virtual ~tPersistent() { }
  int Read(FILE *fp);
  int Write(FILE *fp);
  static int ReadAny(FILE *fp,
              tPersistent *&p);
  };  // class tPersistent
/////////////////////////////////////////
#include  "tpersist.h"
#include  "dlistreg.h"

tRegistration RegPersistent(
   TPERSISTENT, tPersistent::build);
```

```
int tPersistent::Write(FILE *fp)
  {
  int id = ID();
  fwrite(&id, sizeof(int), 1, fp);
  return __write(fp);
  }  // tPersistent::Write()

int tPersistent::Read(FILE *fp)
  {  // assumes object already built and
  // that the object to read *is* of
  // same class as object in memory
  int id;
  fread(&id, sizeof(int), 1, fp);
  if (id != ID()) return 0; // error
  return __read(fp);  // get data
  }  // tPersistent::Read()

int tPersistent::ReadAny(FILE *fp,
           tPersistent *&p)
  {  // looks at ID in file to det.
     // what object will be read
  int id;
  fread(&id, sizeof(int), 1, fp);
  tBuildFunc b;
  b = Registrations->MatchID(id);
  if (!b) return 0;             // error
  p = b();          // create empty object
  return p->__read(fp);  // read its data
  }  // tPersistent::ReadAny()
// end of tpersist.cpp
```

Figure 1 - class `tPersistent`

BE PERSISTENT

1992 BYTE AWARD OF DISTINCTION

COMPUTER LANGUAGE PRODUCT EXCELLENCE AWARD 1991 JOLT COLA

POET PERSONAL EDITION FOR WINDOWS £99.95* CALL FOR DETAILS

Across Platforms

True cross platform support is available from only one ODBMS, and no wonder it is the best selling ODBMS. POET is scalable across DOS, Windows, Mac, OS/2, and UNIX. All POET servers support these platforms in heterogeneous networks. Portability is built in so you can leverage your C++ programming investment across every major C++ compiler.

POET is a true ODBMS that supports encapsulation, inheritance, polymorphism, and complex objects with pointer references and container classes. POET automatically gives you objects methods for disk storage and retrieval, and query classes for value based and identity based queries. New administrative tools allow you to analyze, recover, reorganize, and convert databases.

Our new client/server architecture uses event handling to keep clients shared objects concurrent. Callback functions and our watch/notify API give you the technology needed for tomorrow's workgroup applications. POET features also include variable length strings and BLOBs, simple and nested transactions, two-phased commit and rollback, flexible object locking, optimized parallel queries, and referential integrity.

POET - The only true cross platform ODBMS

Object Store	UNIX		Windows			
Versant	UNIX					OS/2
Objectivity/DB	UNIX					
GemStone	UNIX				MAC	
POET	**UNIX**	**DOS**	**Windows**	**NT**	**MAC**	**OS/2**

POET is available in both Personal (single user only) and Professional (client/server SDK) Editions.

CALL NOW and find out more about POET, the best selling ODBMS.

Silicon River

081 317 7777

58-60 Beresford St
LONDON, SE18 6BG
Tel: (+44) 81 317 7777
Fax: (+44) 81 316 7778

All trademarks are the property of their respective holders

➤ CIRCLE NO. 156

look at the __read() and __write() functions defined first in class tPersistent of Figure 1, you will see that later, they have been redefined in derived classes tPoint and tCircle as shown in Figure 2. tPersistent, by the way, is the base class from which all persistent classes will be derived (either directly or indirectly).

With this basic persistence, you have the capability to write objects to a file and later read them back (during the same or a subsequent run) *so long as you know in advance exactly what object is at which position in the file.* The problem, of course, is that if you're writing only the objects' data to the file, nothing exists later to give you a clue about the original class of the objects. The format of the file must be hard-coded into the source of your program. This isn't as configurable or free-wheeling as we want. As a matter of fact, I would go as far as stating that its plain bad design.

Polymorphism Needs

For example, assume you were writing a drawing program which could display any number of circles, lines, points, or curves on the canvas. The number of objects of each class, and their relation' to each other, is completely dependent on the user's slightest whim. Obviously we need some kind of key in the file to let us know the class of each object before it is read. That is the purpose of tPersistent::ID(). ID() is a virtual function redefined for each new class to return a unique integer key for objects of that class. That number is stored in the file just prior to each object's data, giving us a chance to decide which __read() function to call.

Polymorphic Write

We could simply write the object's ID to the file as the first step of each __write() function, but this would mean that derived objects would store the ID for their base classes as well as their own ID - a bit redundant. To save space, it works much better to have a master Write() function (as in tPersistent) that writes only the ID for the *actual* class of the object, then starts up the chain of __write() calls to store the object's data. For example, when a tCircle is written, it will show up on disk as a) the _TCIRCLE ID number (3), b) the rad data member of tCircle, and c) the row and col data members of tPoint. Had we written the ID number directly in __write() we would also have stored an ID for tPoint and for tPersistent, rather a waste of space considering that, once you know the object is a tCircle, you know for sure that it is also a tPoint and a tPersistent.

Again, note that, since ID() and __write() are declared as virtual, Write() need only be defined once in the base tPersistent class.

Notice that when you derive a new class you should redefine __write(), but you should never call it from your application. The rule to remember is: 'define __write(), call Write()'. The only exception is that inside the __write() function itself you must call the __write() function of the base class.

Polymorphic Read

Now let's think about how we would read these objects from disk later. The sequence of events should be: a) read the ID, b) build an empty object of the class indicated by the ID and c) tell the empty object to read itself. What is wrong with this? Well, for starters, if Read() is going to be a member function (like Write()) the object must be built before you start. So let's make a Read() function like that - assuming the object has already been built, it simply reads the ID, makes sure it matches, then calls __read()

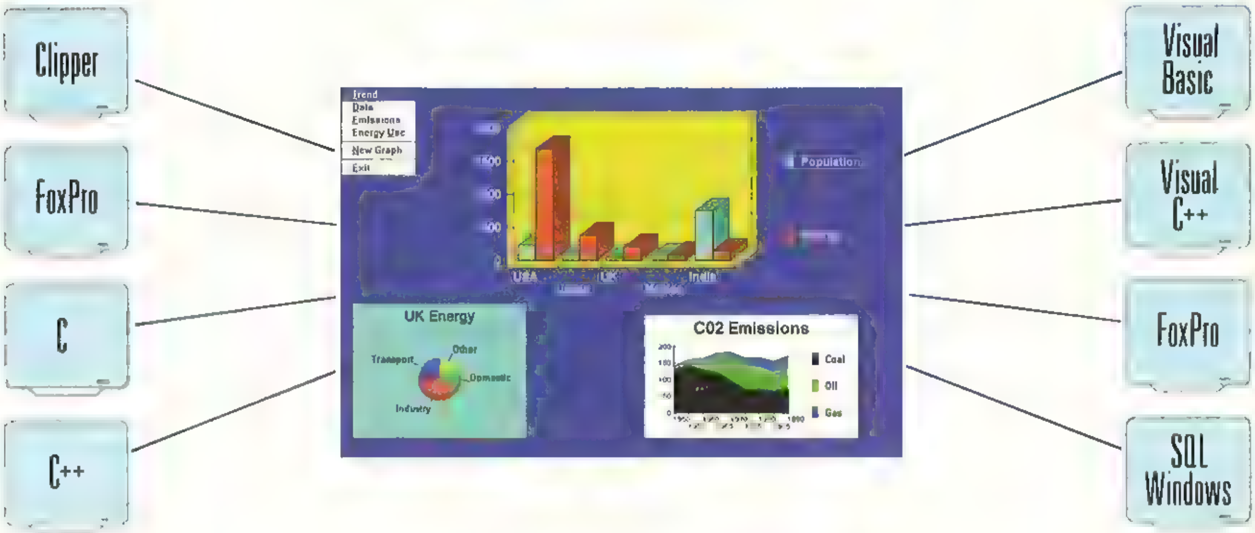
```
// TEST.CPP
#include  "tpersist.h"

// a persistent point
const int _TPOINT = 2;
class tPoint : public tPersistent
  {
protected:
  int row, col;
public:
  tPoint(int r = 0, int c = 0)
    { row = r; col = c; }
  virtual void Print()
    { printf("tPoint: %d, %d\n",
           row, col); }
  // from here down for persistence
protected:
  int __read(FILE *fp);
  int __write(FILE *fp);
public:
  static tPersistent *build()
    { return new tPoint; }
  int ID() { return _TPOINT; }
  };  // class tPoint

tRegistration RegPoint(
   _TPOINT, tPoint::build);

int tPoint::__read(FILE *fp)
  {
  if (fread(&row, sizeof(row), 1, fp)!=1)
    return 0;
  if (fread(&col, sizeof(col), 1, fp)!=1)
    return 0;
  return 1;  // ok (not EOF)
  }  // tPoint::__read()
```

```
int tPoint::__write(FILE *fp)
  {
  if (fwrite(&row, sizeof(row), 1, fp)!=1)
    return 0;
  if (fwrite(&col, sizeof(col), 1, fp)!=1)
    return 0;
  return 1;  // ok (no write error)
  }  // tPoint::__read()


// another persistent class derived from tPoint
const int _TCIRCLE = 3;
class tCircle : public tPoint
  {
protected:
  int rad;
public:
  tCircle(int r=0, int c=0, int ra=0)
    : tPoint(r, c) { rad = ra; }
  void Print()
    { printf("tCircle: %d, %d, %d\n",
           row, col, rad); }
  // from here down for persistence
protected:
  int __read(FILE *fp);
  int __write(FILE *fp);
public:
  static tPersistent *build()
    { return new tCircle; }
  int ID() { return _TCIRCLE; }
  };  // class tCircle

tRegistration RegCircle(
   _TCIRCLE, tCircle::build);

int tCircle::__read(FILE *fp)
  {
```

```
  if (fread(&rad, sizeof(rad), 1, fp)!=1)
    return 0;
  return tPoint::__read(fp);
  }  // tCircle::__read()

int tCircle::__write(FILE *fp)
  {
  if (fwrite(&rad, sizeof(rad), 1, fp)!=1)
    return 0;
  return tPoint::__write(fp);
  }  // tCircle::__write()


int main()
  {
  // create a couple test objects
  tPoint *p = new tPoint(25,30);
  tCircle *c = new tCircle(15, 40, 50);
  // save to a file
  FILE *objs = fopen("OBJS.DAT","wb");
  p->Write(objs);
  c->Write(objs);
  fclose(objs);
  delete p; delete c;
  // now read into tPoint* and print
  objs = fopen("OBJS.DAT","rb");
  tPoint *ptr;
  tPersistent::ReadAny(objs,(tPersistent *)ptr);
  ptr->Print();
  delete ptr;
  tPersistent::ReadAny(objs,(tPersistent *)ptr);
  ptr->Print();
  delete ptr;
  return 0;
  }
// end of test.cpp
```

Figure 2 - test program with example tPersistent *derivatives*

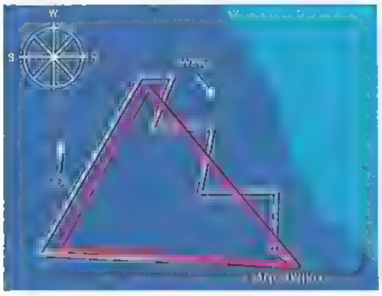
Programmable Graphs



Whatever language you have chosen for DOS or Windows development, you can use one of our graphing libraries to integrate graphs and charts into your application

★1993★ DATABASED ADVISOR Readers Choice Award

dGE has won the 1993 vote as **Best Graphics Tool,** (six years in a row!). Last year Graphics Server won **Best DLL.**



You can create the full range of business and scientific graphs with just a few lines of 'AutoGraph' code, or control every aspect of your design with the 150 functions in the low-level API.

> **'Graphics Server is the most sophisticated of these programs and would be my choice for general use'**
>
> *review of five graphing programs in Windows Tech Journal April '93*

GRAPHICS SERVER

- **'Hot' charts for Executive Information Systems**
- **Includes enhanced graph control for VB and VC++**
- **Works with Ingres/Win4GL and Paradox as well**
- **dGE code will run virtually unmodified in Graphics Server**

WINDOWS NT and OLE 2

- **New – Windows NT version of Graphics Server**
- **Super-fast real-time graphing and charting**
- **New – OLE 2.0 Automation object**
- **The beginning of the end for function calls!**

PS. dGE code will run virtually unmodified under Windows with Graphics Server.

> **"You won't find an easier way to add charting to your Windows applications."**
>
> *PC Magazine, September 1993*

We founded Bits Per Second ten years ago, when getting graphs into a dBASE app was the leading edge of PC development.

We've stayed out in front and now count Microsoft and Gupta among those who license our programmable graphs.

Windows NT will open up new ways of visualising data, such as real-time charting.

If you'd like to hear more about our experience with NT and OLE automation, and about our plans for 1994, give us a call.

John Marsh G Knott

John Marsh Graeme Knott

Graphics Server for Windows and Windows NT is £395. For Windows only, £245. It includes Chartbuilder graph control, available separately for £85. dGE is £245. Our products come with a 60-day money-back guarantee.

Bits Per Second, 14 Regent Hill, Brighton,
East Sussex, BN1 3ED. Tel: 0273 727119

BITS PER SECOND

Bits Per Second recognise that all company and product names are the property of their respective holders.

➤ CIRCLE NO. 157

to get the object's data. While this has the single advantage over simple persistence of warning you if you try to read a stored object into the wrong class of object in memory, it still requires that you know the file structure in advance (simply telling you that you screwed up, but not correcting for it). Even more significant, since the main program must know every possible class of persistent object to be able to build the empty objects, any new class of object added to the program will require modification to the main program to detect it (probably in the form of several gigantic switch statements). Again, bad design.

Back to our drawing program example, when do we read the ID? If we first built an empty tPoint, told it to read itself, and it then discovered an ID for tCircle in the file, what would it do with the extra data? For that matter, how can the tPoint object we've built call the __read() function for another class that it doesn't even know about? I think you can see the Catch 22 here. Obviously, we can't rely on a member function to read objects if we don't know in advance of what class they will be. That's the reason Read() protects us against this by checking the ID of the object on disk and returning zero (failure) if it is different to the ID of the object in memory.

Class Registration

In order to allow fully polymorphic reading of objects from disk, we need two features: 1) a static member function (one that doesn't need an object in existence prior to being called) which follows the a, b, c steps given above, and 2) a list containing information for each persistent class that the program may encounter. The function is tPersistent::ReadAny(). This uses the list Registrations to search for a matching ID and calls a build() function which creates a new empty object of the corresponding class, after which it tells the new object to __read() itself.

Slow down. Things are getting a bit thick and deep at the same time here! Let's attempt to explain. Each descendant of tPersistent has a member function called build(), which creates a new object of the given class. (Note that build() functions are always declared static, so we don't need an existing object to call them). For every persistent class, a global object of class tRegistration (Figure 3) is declared. A tRegistration keeps its related tPersistent class' build() address as well as its ID number. Not only that, but when a tRegistration is created, its constructor automatically adds it to the list of all classes registered in the program (Registrations). When the application later wants to read a persistent object from disk, it simply calls tPersistent::ReadAny(), sending it a tPersistent*. ReadAny() will then read the ID from the file, search Registrations for a class with matching ID, call that class build() and, pointing to the newly built object with the given tPersistent*, finally call __read() of the new object. Phew!

You must always remember that the order in which global object constructors are called is unspecified in C++

Since all tRegistrations are global, they will be constructed. Therefore the Registrations list will be completed, before execution of the main program starts. This is convenient, as it means you can add new persistent classes without changing the main program to recognise them. It also has a few implications: first, you must always remember that the order in which global object constructors are called is unspecified in C++. That means that, if the Registrations list were also a global object, it might be possible for one of the tRegistration items to be constructed and attempt to add itself to the list before the constructor of the list itself had been called. To eliminate this possibility, the Registrations list is a dynamic object which is created automatically by whichever tRegistration happens to be constructed first (see the constructor of tRegistration). Likewise, as tRegistrations are destructed, they automatically remove themselves from the list. The last one to be destructed also automatically deletes the list.

Another implication of having the Registrations list created by global constructors is that you must *never* attempt to read persistent objects in a global constructor. This is unfortunate, as it means that persistent objects can't be automatically read from disk at the beginning of the program without the main program's knowledge. It's free.

```
#if !defined( TREGISTR_H)
#define __TREGISTR_H

#include <tobject.h>

class tPersistent;
typedef tPersistent *(*tBuildFunc)();

class tRegistration : public tObject
  {
  tBuildFunc build;
  int iD;
public:
  tRegistration(int i, tBuildFunc b);
  ~tRegistration();
  int ID()           { return iD; }
  tBuildFunc Build() { return build; }
  };  // class tRegistration
#endif
////////  /////////////////////////////
#include  <stdio.h>
#include  "tregistr.h"
#include  "dlistreg.h"

tRegistration::tRegistration(int i,
                             tBuildFunc b)
  {
  // check for no list and create if not
  if (!Registrations)
    Registrations
      = new DListtRegistration;
  build = b; iD = i;
  if (!Registrations->Insert(this))
    printf("ERROR: Dup. Registration
%d!\n", i);
  }  // tRegistration::tRegistration()

tRegistration::~tRegistration()
  { // remove from list
  if (Registrations)
    {
    Registrations->Get(this);
    if (Registrations->IsEmpty())
      { // assure list disposed of
      delete Registrations;
      Registrations = 0;
      }
    }
  }  // tRegistration::~tRegistration()

// end of tregistr.cpp
```

Figure 3 - class tRegistration

OST-LOOK! C++ PROGRAM ANIMATION SYSTEM

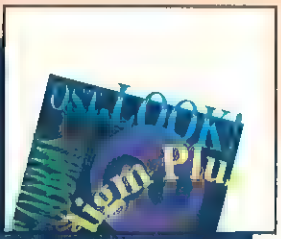
The OST-LOOK! C++ program animation system

OST-LOOK! is the first system that lets you "see inside" your C++ program. OST-LOOK! lets you graphically filter, animate and explore your C++ program ***as it executes.*** It cuts through the complexity of C++ by presenting intuitive graphical views of what is happening. This means that you can see structural problems immediately and understanding new code becomes much easier as OST-LOOK! shows it working - like an animated whiteboard.

What makes program animation possible is a monitoring system which allows OST-LOOK! to dynamically reason about program structures. Using this, OST-LOOK! generates multiple, concurrent graphical views of your program as it runs.

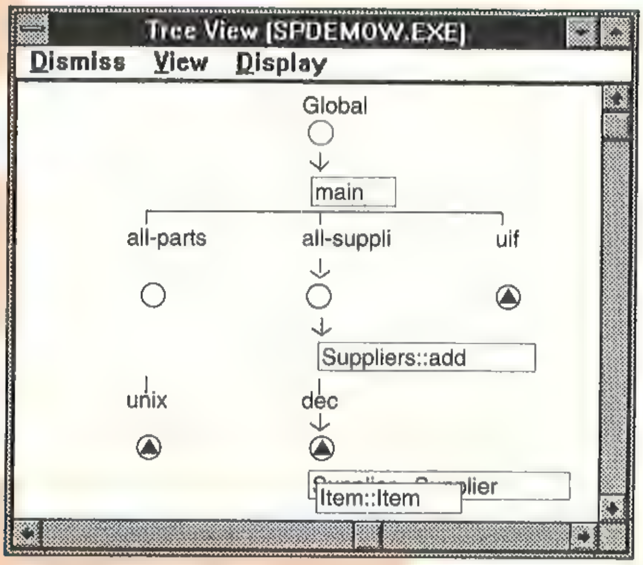
OST-LOOK! Technical Features

- Multiple, concurrent animated views of C++ applications
- Store and replay facilities active on all graphical views
- Object-based dynamic filter language lets you specify the part of the program to visualise in terms of objects, classes, messages and their attributes
- Dynamic modification of filter criteria both directly and by programmed filter changes which trigger as the application executes.



OST-LOOK!

An animated whiteboard for C++ developers



Available on PC and Sun workstations, version 1.0 of OST-LOOK! supports Borland, Microsoft and Sun C++ compilers.

OST-LOOK! was developed in Scotland by Objective Software Technology and is marketed by Admiral Software Limited.

admiral

CALL NOW ON:

0276 692269

or return the reply slip below.

I would like to receive more information on OST-LOOK! .EXE 11/93

Name: ______________________

Job Title: ______________________

Company: ______________________

Address: ______________________

____

Tel No: ______________________

Fax No: ______________________

Admiral Software Limited
Admiral House, 193-199 London Road,Camberley, Surrey GU15 3JT.
Tel: (0276) 692269. Fax: (0276) 677533

➤ CIRCLE NO. 158

What do you expect?

The class for the list, DListtRegistration, is derived from my tried and trusted DList class which I've been using without incident for nearly three years now. I haven't included it in the printed listings, but it is available with the rest of the code on disk or via ftp.

Although you must manually give an ID number to each new persistent class, DListtRegistration does have a safeguard that warns you if you accidentally give the same ID to two different classes. The Insert() function in Figure 4 checks before adding a new registration, printing an error if the same ID is already in the list. Although the error message here is quite ugly, it would only occur during program development and would happen either every time the program was run or never: it wouldn't mysteriously crop up on an irate client's screen some day six months from now. This 'all or nothing' character made me comfortable with the relative primitiveness of the error catching scheme.

New Classes

The example in Figure 2 shows what you must do to derive a new tPersistent class, as well as demonstrating that the scheme actually works. To preserve the persistence in a new class, you must:

1) derive it directly or indirectly from tPersistent. Multiple inheritance is allowed, but you must remember to take care of all base classes in __read() and __write().

2) define a static build() function which creates a new object of the class and returns it as a tPersistent* (necessary to avoid type mismatch problems).

3) define a new, unique ID constant and an ID() member function which returns that value.

4) define __read() and __write() functions which read or write any data newly declared in this class, then call __read() or __write() of the base class

If Bjarne had meant us to program only using built-in features of C++, he would have disallowed class definitions and included MFC V2.0

5) declare a global object of class tRegistration, sending it the ID number and build() address for the new class.

Anything in the new classes beyond these five features is completely independent of its persistence.

Using Persistence

The main() function in Figure 2 gives primitive examples of using the Write() and ReadAny() functions. Of particular interest is the use of ReadAny(). Since it is a static member function, we don't need an object to call it. To inform the compiler that we want 'ReadAny() - the member of tPersistent' rather than 'ReadAny() - the global function', we must qualify the name, though: hence the 'tPersistent::ReadAny()'. Also, notice that I had to typecast the tPoint* to tPersistent*. This is due to C++'s desire to create a temporary variable of type tPersistent* initialised to the value of tPoint* rather than using the original and doing the typecast itself.

```
#include  "dlistreg.h"

DListtRegistration *Registrations = 0;

tBuildFunc
DListtRegistration::MatchID(int i)
  { // find build function for i
  GotoHead();
  while(Current())
    {
    if (Current()->ID() == i)
      return Current()->Build();
    Next();
    }
  return 0;
  }  // DListtRegistration::MatchID()

int
DListtRegistration::Insert(tRegistration *s)
  {  // insert, checking for duplicate
  if (MatchID(s->ID()))
    return 0; // ERROR - duplicate ID
  ToTail(s);
  return 1;
  }  // DListtRegistration::Insert()

// end of dlistreg.cpp
```

Figure 4 -
Functions newly defined in class DListtRegistration

C++ does this, by the way, to avoid the following situation: a) tPoint* is sent as reference parameter to a function asking for tPersistent*, b) the function actually makes the tPoint* point to a tPersistent and c) after return, the calling function attempts to call a tPoint function on the tPersistent object. Oops! In our case, however, we know that we will only get back tPoints or tPoint derivatives from the call, so we can safely typecast.

And Modula freaks tell you that C++ is loosely typed and dangerous...

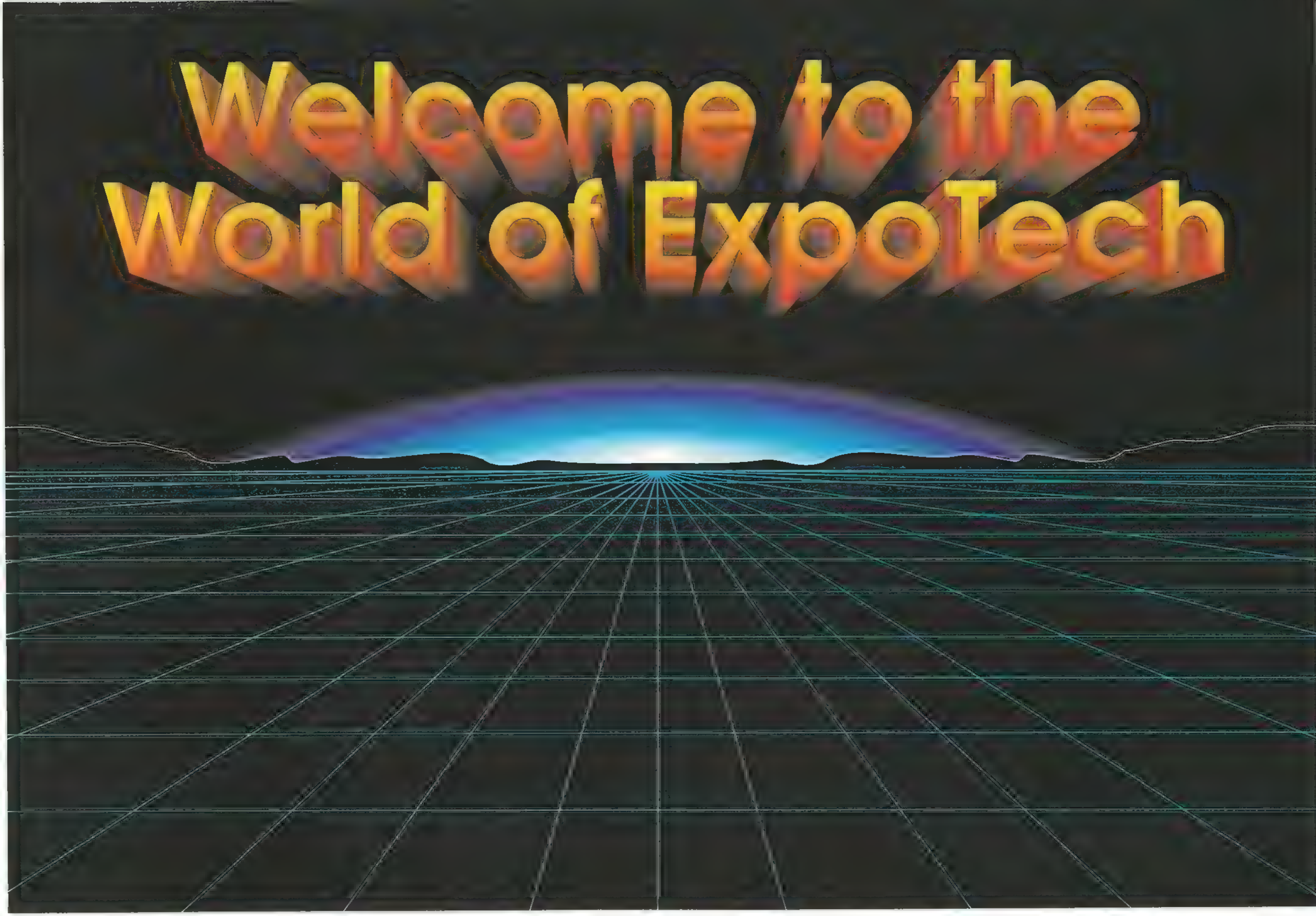
Summary

Classes can be given persistence capability with the addition of a few simple utility classes to your library and an additional four short function definitions for each class. A registration list containing information for each persistent class is essential to allow polymorphic reading of objects.

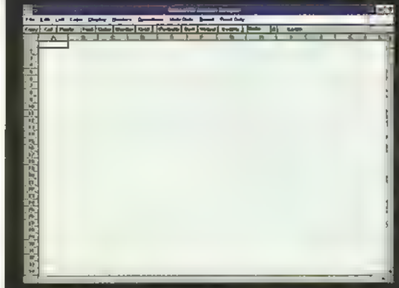
Although the persistence method given here is not as all-encompassing as commercial packages, it is workable in simple situations and serves as a good example to help you understand how polymorphic persistence works.

EXE

Laine Stump is a software engineer turned university OOP instructor turned ??? Although he recently moved to Columbus, Ohio, you can still reach him by email through Bilkent University in Ankara, Turkey: laine@bilkent.edu.tr. Sample code from Laine's articles is usually available via anonymous ftp from firat.bcc.bilkent.edu.tr in the directory pub/Local/Cpp. To make this month's example work, you'll need the files gebui.zip and persist.zip.


Welcome to the
World of ExpoTech

Spread/VBX v2.0



Spread/VBX, originally known as Visual Architect, includes a spreadsheet custom control that gives sophisticated grid capability. Spread/VBX provides a set of tools necessary to Visual Basic developers. All kinds of data input can be handled, no matter what type of application is being developed. Minimal coding effort is required. The speadsheet control has strong flexibility and power. It is able to be used within the application as a functional spreadsheet, as an easy way of obtaining variable lines of data from the user, or as a simple way of displaying tables of information from a database. Simpy put... Spread/VBX is the ultimate spreadsheet control.

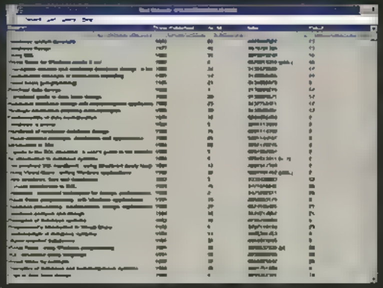
➤ CIRCLE NO. 150

Compatibility

▲ Visual Basic v3
▲ Visual C++ v1

➤£195◄

TrueGrid v1.0



TrueGrid is a fully editable data-aware table control for Visual Basic v3. This very easy-to-use control works seamlessly with VB3's data-access capabilities. This technology is derived from the data-grid Apex provides with its Agility database product, so it has been tested by thousands of database application developers.

In addition Apex have thoroughly enhanced it's capabilities to satisfy the peculiarities of database usage, with features like multiple selection and customizable appearance. These designs combined with its ease-of-use design, proven technology and price make this a must have product.

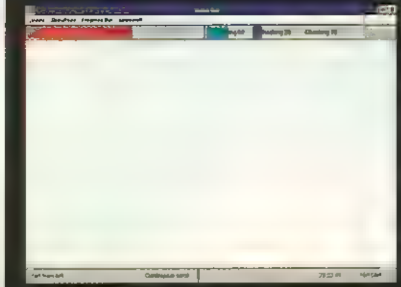
➤ CIRCLE NO. 185

Compatibility

▲ Visual Basic v3

➤£69.95◄

ToolsKan/VB



ToolsKan/VB is a collection of five VBX files. The Chart provides a 3D or 2D representation of data, which includes all data, labels, axes, grid lines, and legends. This can be a regular chart that accesses data from memory or it can be treated as a database front-end that maintains a cache of a range of current data, fetching data from the application only when needed. The Ribbon is a horizontal bar that integrates combo boxes, edit controls, text strings, and buttons. The Status Bar is comprised of elements for displaying text, and progress meters. The Table is used for displaying data from databases or from memory. The Toolbox control is used to create tool-palettes.

➤ CIRCLE NO. 160

Compatibility

▲ Visual Basic v1
▲ Visual Basic v2
▲ Visual Basic v3
▲ Visual C++ v1

➤£129.95◄

Agility

Agility/VB is a database library specifically designed for Microsoft Visual Basic. Agility combines a small, effective command set with powerful features such as database independence, QBE query processing with query optimization, automatic index selection, and relational views. Powerful custom controls provide sophisticated user-interface power almost instantly without programming. Agility supports dBASE files, ASCII text, and a native format which furnishes added power and flexibility. Multi-user support and additional file formats are on the way. Agility's powerful grid gives complete control.

Compatibility

▲ Visual Basic v1
▲ Visual Basic v2
▲ Visual Basic v3
▲ Visual C++ v1

➤£149.95◄

Also available Agility/VC++

➤ CIRCLE NO. 186

Image Format/VBX

The AccuSoft Image Format Library/VBX is a fully object-oriented Visual Basic & Visual C++ Custom Control. The design of this product takes full advantage of the ease-of-use aspects of object-oriented and visual programming. This Library is so easy to use that raster image support will be able to be added for 10 file formats complete with display, zooming, scrolling and image processing in less than 30 minutes. There are no function calls in this library since everything is implemented as properties of the Custom Control. The Accusoft Image Format Library supports GIF, TIFF, PCX, DCX, WPG, TARGA, BMP, EPS, WMF, PCT, and JPEG.

Compatibility

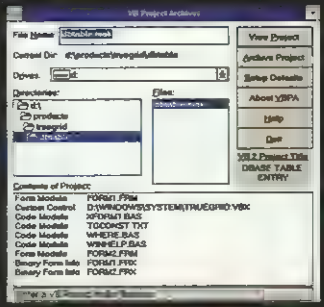
- ▲ Visual Basic v1
- ▲ Visual Basic v2
- ▲ Visual Basic v3
- ▲ Visual C++ v1

➤£345◀

DLL Version also available!

➤ CIRCLE NO. 162

VB Project Archiver



Form Module FORM1.FRM
Custom Control D:\WINDOWS\SYSTEM\TRUEGRID.VBX
Code Module XFORM1.BAS
Code Module TODOAST.TXT
Code Module WHERE.BAS
Code Module WINHELP.BAS
Form Module FORM2.FRM
Binary Form Info FORM1.FRX
Binary Form Info FORM2.FRX

VB Project Archiver is a utility for Visual Basic programmers, which allows the user to store all of the modules and files relating to the VB project in a single file using one of the popular archiving programs (PKZip, ARJ, LHA, ZOO and others), without leaving the comfort of the Windows Environment. VBPA uses the Makefile to get a list of modules and forms used in the project, so there is no need to worry about forgetting anything. VBPA will also scan the Global and Code modules for Declare statements for Windows DLLs or VBXs that might be used within the project! VBPA can even replace design-time Custom-controls with their run-time counterparts.

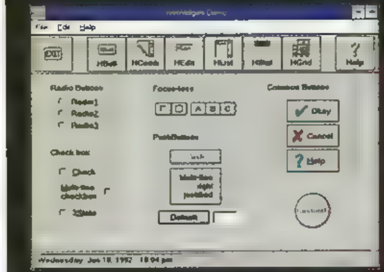
Compatibility

- ▲ Visual Basic v1
- ▲ Visual Basic v2
- ▲ Visual Basic v3

➤£79.95◀

➤ CIRCLE NO. 188

WinWidgets / VB



WinWidgets/VB brings the WinWidgets to programmers in Visual Basic. This edition includes the same robust functionality as the original DLL plus 'Data Awareness' enhancements for use with VB v3 that make these tools more powerful and easier to use than ever before. With WinWidgets Grid Control and VB v3 you can present your users with a buffered, editable data table that automatically updates your database without writing a single line of code!

Also Available: WinWidgets v2 Professional for C/C++ programmers. Included is support for MFC and OWL. Both products come complete with Hypertext documentation.

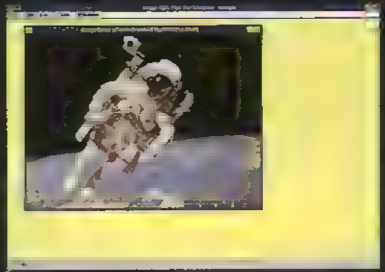
Compatibility

- ▲ Visual Basic v1
- ▲ Visual Basic v2
- ▲ Visual Basic v3

➤£145◀

➤ CIRCLE NO. 187

Image SDK Plus



The New Image SDK Plus has arrived!! Read/Write/Print TIFF, PCX, GIF, TARGA, MS-Metafile, Clipboard, DIB and ColorFax Format. Image transformation: Antialiasing conversion, scaling colour images, bilinear or cubic spline interpolation, skew image for OCR, rotation in 1 degree increments, dithering, display DIB or DDB image, compress and decompress DIB, scrolling, utility functions, digitize ASCII text, utility functions, digitize ASCII text.

The package is supplied compete with manuals and example code - No royalties.

Compatibility

- ▲ Visual Basic v1/2/3
- ▲ Borland C++
- ▲ Microsoft C v7
- ▲ Visual C++ v1

➤£249.95◀

➤ CIRCLE NO. 189

Dyad M++ v5

M++

M++ provides a multi dimensional (up to 4 dimensions) array language extension to C++. M++ includes both LINPACK linear system classes and EISPACK eigensystem classes and allows the user to perform array manipulations, numeric operations on arrays and sub-arrays, as easily as on scalars. An important feature of M++ is the ease with which users of APL, MATLIB, GAUSS and other array languages can quickly write code in a similar array program style. The M++ Class Library provides methods and functions for array, matrix and vector manipulations as well as a full complement of linear system and eigensystem analysis classes. New M++ Modules including LSM, ODE, QUAD, OPTIM, SUM, VIS, TEST and FEM.

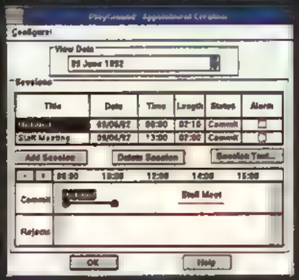
Compatibility

- ▲ Borland C++
- ▲ Microsoft C

> CIRCLE NO. 161

➤£295◀

Out of Controls



Out of Controls is a set of 5 Custom controls for use within Windows. The controls include: Spin Box Control with full function edit box; Modal Spawning Library for opening DOS/Windows; Multi Cell List Box - rows may contain multiple cells of any type, automatically scrolls vertically and horizontally, allows rows to be reordered by drag and drop; Text Editor Library adds an editor to a program in seconds, supports files up to 32k in size; Line Object Editor in which each object may be configured as either mixed or movable against the ruler, each object may be configured as fixed size, sizeable or split and rejoinable, allows mutliple category rows for sorting objects and also supports customized ruler formats.

Compatibility

- ▲ Borland C++
- ▲ Microsoft C

> CIRCLE NO. 191

➤£195◀

Drovers Toolbox v3

With 23 custom controls, including FarPoints industry unique full-featured spreadsheet control (not a grid!), Formatted Editor Controls, Tool Bar and Status Bar, ability to add 3D effects to dialog boxes, View Pictures with animation, and Enhanced Listbox this package is a developer's dream come true! Over 300 functions are built in, including DOS System functions, Data/Time support, String Functions and Enhanced file support.

Drover's Toolbox is compatible with Dialog Editor, Resource Workshop, Actor, WindowsMAKER.

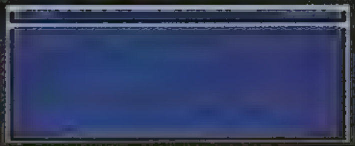
Compatibility

- ▲ Borland C++ V2/3
- ▲ Microsoft C
- ▲ Actor

> CIRCLE NO. 190

➤£295◀

EZ-Install



EZ-Install produces professional comprehensive installation routines for software products. Includes menu-driven Installation Aid, which steps the developer through the installation option, generates the installation configuration, then creates distribution disk(s). Checks/modifies user's CONFIG.SYS, AUTOEXEC.BAT, Windows .INI files, and system environment. Compression, File-splitting utilities and screen builder utilities are included.

EZ-Install has NO copyright message on the runtime programs, NO copy protection and NO royalties payable on distributed installation programs.

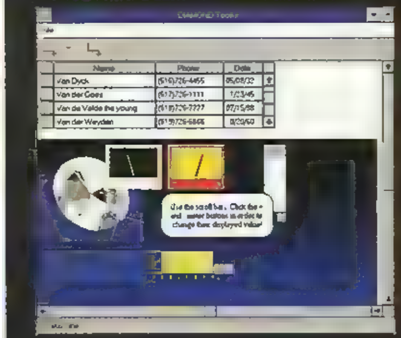
Compatibility

- ▲ DOS/Windows
- ▲ Optional OS/2

> CIRCLE NO. 192

➤£195◀

Diamond Toolbox



Compatibility

- ▲ Borland C++ v2/3
- ▲ Microsoft C v7

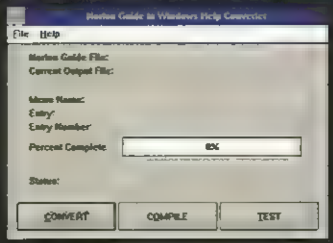
➤ CIRCLE NO. 163

➤£295◀

Diamond Toolbox is seven packages in one: Cockpit, Revision Control, LoadIt, HelpIt, Field Validation, Gadget Library together with the actual toolkit and source code.

Using Cockpit, the advanced workbench and revision control system, you are only three clicks away from your first Windows Application. Cockpit handles resource, make, include, definition and source files. You only have to deal with the functions and projects. LoadIt is a powerful installer that requires no compilations for generating setup configurations. LoadIt supports dialog boxes, paragraph tags, bitmaps, background fountain fills - use it to build simple applications too!

NG-2-HLP



Compatibility

- ▲ Windows v3
- ▲ Help Compiler

➤ CIRCLE NO. 194

➤£69.95◀

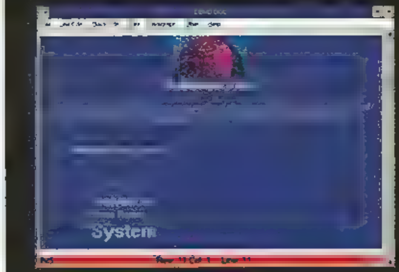
NG-2-HLP is a great new utility that enables Norton Guide users to migrate their DOS based Help Systems to Windows!

Its as easy as 1,2,3!

Just open your existing Norton Guide File, press the Convert Button and then the Compile Button. It's that simple!

NG-2-HLP will start working as soon as its installed - there are no complex commands or confusing configuration options to learn. If you can use Windows you can use NG-2-HLP!

TE Developers Kit



Compatibility

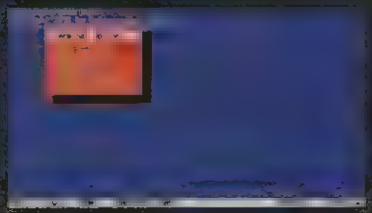
- ▲ Visual Basic v1/2/3
- ▲ Borland C++
- ▲ Microsoft C v7
- ▲ Visual C++ v1

➤£249◀

Incorporate text editing features into your application easily and cost effectively. Features include: multiple files and windows, word-wrap, edit large files, reconfigurable keyboard and screen colours, undo, cut/paste, print, search/replace, word processing features such as WYSIWYG, indentation, double spacing, centering and right justification imbedded pictures, multiple fonts and sizes. The kit also includes TER small editor routine a subset of TE. Your application calls this routing by specifying the window location and size, maximum file size and an input buffer or file. Includes the complete 'C' source code.

➤ CIRCLE NO. 193

Telepathy



Compatibility

- ▲ Clipper V5

➤ CIRCLE NO. 195

➤£195◀

Telepathy is the most powerful and complete serial communications library ever developed for Clipper. Telepathy makes it easy to add the communication functions you've always wanted in your Clipper applications. For optimal performance, a powerful 'notification mechanism allows Telepathy to process data in the background and supports the direct use of serial devices such as bar-code wands in place of the keyboard. Telepathy supports the most popular binary file transfer protocols: Xmodem (and Xmodem-CRC and Xmodem-1K), batch-modem Ymodem and Ymodem-G, Kermit, and Zmodem. Progress functions are one of Telepathy's many unique features.

Recommended Retail Prices

Accusoft Image Format Library v4	£345.00
Accusoft Image Format Library/VBX v4	£345.00
Apex Agility/VB	£149.95
Apex Agility/VB with Crystal Reports	£195.00
Apex Agility/VC	£149.95
Apex Agility/VC with Crystal Reports	£195.00
Apex TrueGrid	£69.95
Black Ice TIFF SDK for Windows	£249.95
Black Ice TIFF SDK for DOS	£249.95
Black Ice Image SDK Plus for Windows	£249.95
Black Ice Targa SDK for Windows	£99.95
Black Ice GIF SDK for Windows	£99.95
Black Ice PCX/DCX SDK for Windows	£99.95
Celect Out of Controls v1	£195.00
Celect Out of Controls v1 with Source	£345.00
Dyad M++ v5 (DOS)	£295.00
Dyad M++ v5 with Source (DOS)	£595.00
Dyad M++ v5 (OS/2 or NT)	£495.00
Dyad M++ v5 with Source (OS/2 or NT)	£795.00
Dyad M++ v5 (UNIX)	£645.00
Dyad M++ v5 with Source (UNIX)	£995.00
Extrasensory Telepathy	£195.00
Extrasensory Faxual.lib	£99.95
Extrasensory Sez You	£49.95
Extrasensory PS-Error	£129.95
Extrasensory Postscript Library	£129.95
Farpoint Spread/VBX v2	£195.00
Farpoint Spread/VBX v2 with Source	£595.00
Farpoint Drover's Toolbox v3	£295.00
Farpoint Drover's Toolbox v3 with Source	£995.00
Kansmen ToolsKan/VB	£129.95
Kansmen Toolbox	£129.95
Kansmen Status Bar	£129.95
Kansmen Field Validation	£129.95
Kansmen Ribbon/Icon Bar	£129.95
Kansmen Meter/Gauge	£129.95
Kansmen 3D Chart	£349.00
Kansmen Table	£349.00
Klondike Diamond Toolbox V1	£295.00
RAD Systems R.A.D.	£295.00
Simple WinWidgets/VB	£145.00
Simple WinWidgets Pro	£295.00

Simple WinWidgets Pro with Source	£595.00
Sub-Systems TE Developers Kit for DOS v3.5	£219.00
Sub-Systems TE Developers Kit for Windows v3.5	£249.00
Sub-Systems TE Developers Kit for OS/2 v3.5	£249.00
Sub-Systems ReportEase for DOS v1.5	£249.00
Sub-Systems ReportEase for Windows v1.5	£269.00
Sub-Systems Spell Time for DOS v1.0	£249.00
Sub-Systems Spell Time for Windows v1.5	£249.00
Sub-Systems Spell Time for OS/2 v1.5	£249.00
Sub-Systems ChartPro for Windows v1.5	£269.00
Software Factory EZ-Install	£195.00
Software Factory EZ-Install (DOS/OS2)	£249.00
Urquhart Tremayne CAESAR	£495.00
Urquhart Tremayne HAT	£295.00
Urquhart Tremayne NG-2-HLP	£69.95
Young Software Works - VBPA	£79.95

Prices shown exclude carriage & VAT

For product information or details of your nearest stockest, please contact our Customer Services department on the number shown below.

➤ CIRCLE NO. 164

ExpoTech
CORPORATION

The Leaders in Development Tool Distribution

ExpoTech House, Western Road, Hove, Sussex. BN3 1AF
Sales: +44-(0)273-749222 Fax: +44-(0)273-746446

Logging on to Paradox for Windows

Paradox, Borland's heavyweight database contender, has generated much interest. John Braga discusses its brand-new development language - ObjectPAL...

Paradox for Windows (P4W to its friends) is a fascinating and maddening product. It has an impressive GUI, powerful database features and a masterly understanding of data relationships. It also has ObjectPAL, a language with very little in common with PAL as used in DOS Paradox. The CompuServe P4W Forum is currently echoing with the tortured screams of developers coming to terms with Windows event-driven programming, an object-based language and the P4W event model all at once...

P4W displays the paradox (sorry!) that doing complicated tasks is simple, while doing simple tasks can seem hideously complicated. So creating a form to show four tables, related in a 1 -> Many -> Many -> 1 relationship (for example Customers -> Orders -> Line Items -> Stock Items) is child's play, whereas setting up an application which uses three forms and passes parameters between them can take two days of frustrating trial and error as you are forced down ObjectPAL paths you have not previously used.

When designing an application recently, I came across the need to control user access by means of passwords. Since this is a frequent requirement, I created a P4W form that is general purpose. In doing so, I found I had to experiment with several ObjectPAL techniques that will be of benefit in many other situations.

Doing complicated tasks is simple, while doing simple tasks can seem hideously complicated

The User Table

I first created a table called USERS to hold names, IDs and passwords. It makes sense to create your tables before designing your forms: you will certainly want to do both before writing a line of ObjectPAL. With such a large and complex beast, positively dripping with features, you will benefit by pointing and clicking as long as possible rather than rushing into writing yards of code.

The USERS table structure is shown in Figure 1. When creating it, make full use of the features to add a picture to each field. For instance use:

```
*!
```

to make all characters upper case (for the ID and Password field)

and

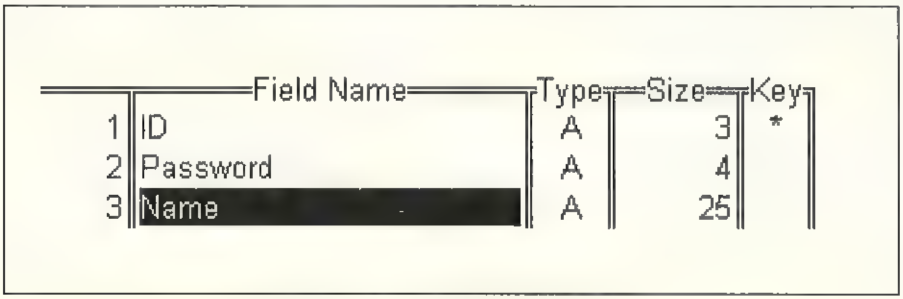
```
!*@
```

to specify that the first character is in upper case (for the Name field). Knowing that the ID and Password will always be in upper case saves a validation step later.

In addition, you should check the Required Field button for all three fields. Such a table is obviously sensitive. To prevent a user opening the table and noting all the passwords, use the Password Security feature to protect it. In this example I added a master password of 'boss' to the USERS table. The table is now protected against access, whether interactive or via ObjectPAL, from all users who do not know the password.

Save the table as PWORD.

Note that, for a full-blown application, routines will needed to be provided to add and delete users, and to change passwords. However I will not be covering these in this article.



	Field Name	Type	Size	Key
1	ID	A	3	*
2	Password	A	4	
3	Name	A	25	

Figure 1 - The USERS Table

The Password Dialog

The dialog used to handle user login is shown in Figure 2. It asks for an

ID and a password. When the user presses the `Confirm` button (or presses ENTER), the dialog opens and searches the USERS table. If the ID is found and the password correct, the dialog returns the name of the user to the calling routine.

The password is not displayed and is not available to the calling routine. Provided that you keep the source of the form to yourself, so that users cannot examine the source code, the data should be safe from hackers.

In P4W almost everything displayed on the screen is an object. The application is built by setting the properties of the objects so that they react in the desired way and by adding code to the built-in methods which are executed when events occur, such as a user pressing a key or clicking with the mouse.

When creating your password dialog, inspect the form (by right-clicking on the title bar) and use the `Window Style...` parameter to:

- Change the Window Style from Window to Dialog.
- Set buttons for Maximise, Minimise and Control Menu OFF (unchecked).
- Set the Frame style to `Dialog Frame`.
- Enter a title such as `Password Entry` or `Logging In`.
- Set the Modal flag `ON` (checked).

You will also want to choose a suitable size for your form. Do this be choosing `Form | Page | Layout`, and then set the Form property to `Size to Fit`. In a dialog of this sort there is no point in letting the user resize the box.

In P4W almost everything displayed on the screen is an object

You need to save the form, close it and reopen it before all these changes are effected. Merely switching from design mode to view data mode will not suffice. The changes allow the user to move the dialog around the screen, but not to change its size or move to other parts of the application (switching to other Windows applications is still enabled).

When creating the password field, set the run-time `no echo` property `ON` (checked) to prevent the password being displayed. If you prefer, you can easily write a few ObjectPAL commands to display asterisks instead of merely moving the cursor. This, as the *Boys' Own Paper* used to say, is left as an exercise for you, Dear Reader...

The Object Tree for the dialog is shown in Figure 3. The underlining shows where user code is attached. In this case small nuggets of ObjectPAL code are attached to the form (the password dialog itself), the two input fields (`ID` and `Password`), and the two buttons (`Confirm` and `Cancel`).

A suitable place to start looking at the ObjectPAL code is the `Pushbutton` method attached to the `Confirm` button (Figure 4). As you would expect, this code gets executed when the user clicks the mouse on `Confirm`.

The logic of the method is simple enough. If either the ID or the Password have been left blank, an error dialog is displayed. If both have been entered, the method relies on the Custom Procedure `lGoodLogin` to determine whether the ID and Password pass muster. If they do, the dialog returns to the calling routine passing the name of the user. (The calling routine can also extract the ID as we will see below.) If either the ID or Password is in error, the method does not return to the caller, so the user has to make corrections.

As you can see from Figure 5, the pushbutton method for the `Cancel` button is shorter.

The `Confirm` and `Cancel` buttons are the only ways by which the called dialog can return to the caller, so if `""` is returned, the user has not entered a valid password.

The validation procedure `lGoodLogIn` is a custom procedure attached to the dialog form. Although, in this case, it could equally well be attached to the `Confirm` button, since it is only referenced within the button. I put a

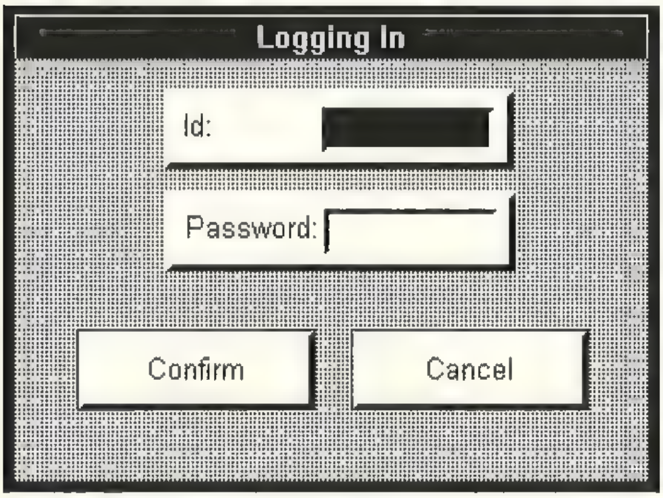


Figure 2 - The Password Dialog

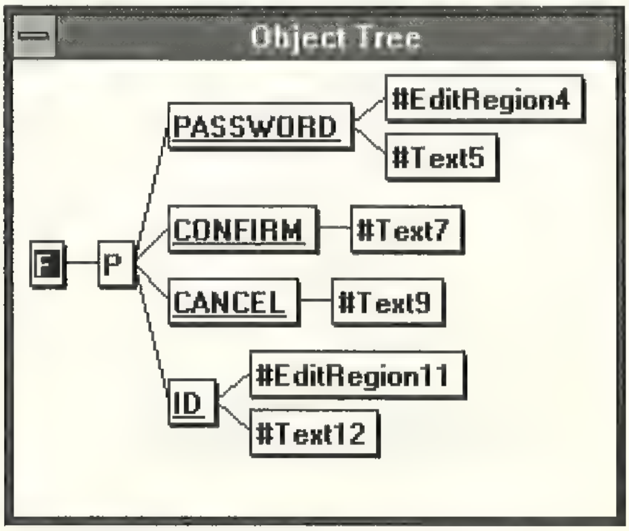
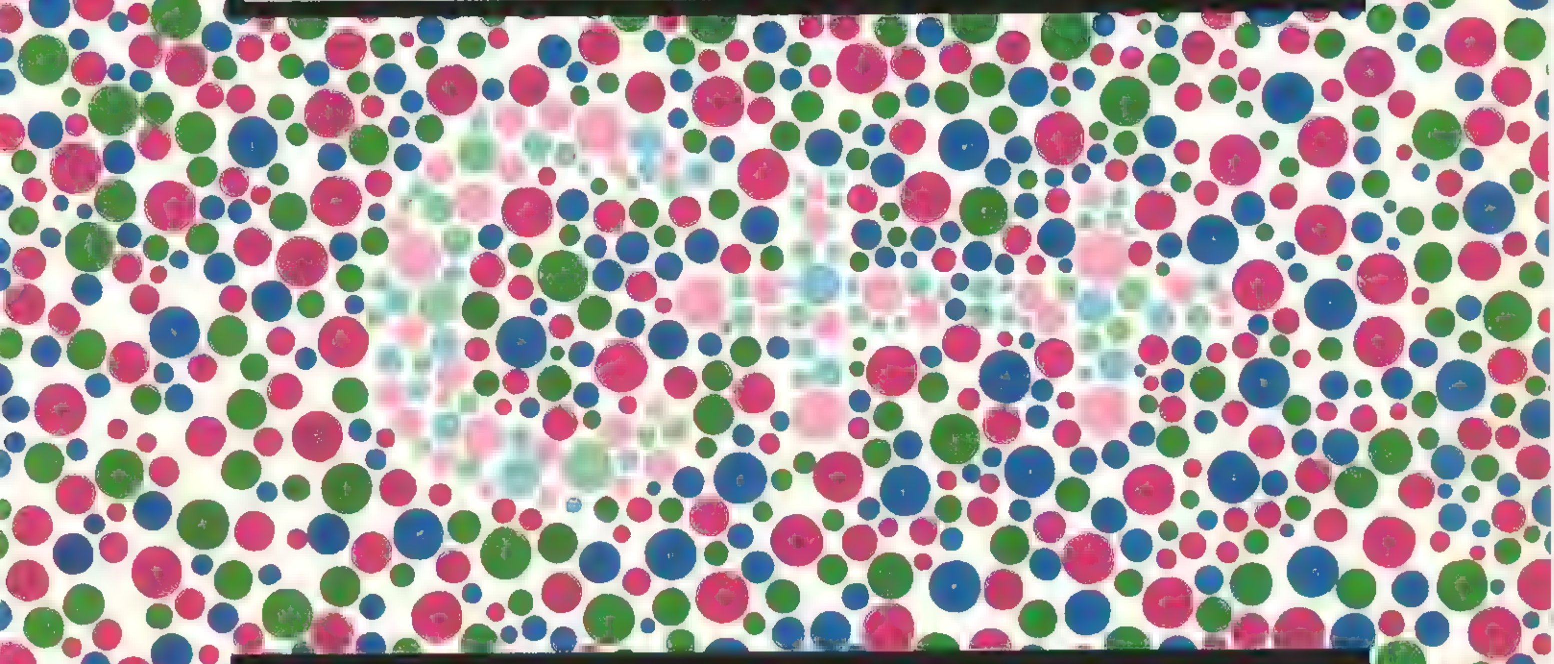


Figure 3 - The Dialog Form Object Tree

HAVE YOU GOT THE VISION?



See your way clear to good design for £195

If you are developing in C++, you'll want to take a very close look at C++ Designer™ – a powerful multi-user Windows based design tool specifically for users of C++. The unique combination of graphical design tool with C++ code generation will save time and effort and help you design better systems!

Adopt the Visual Approach. Instead of struggling with lines of code, C++ Designer lets you visualise the system you are designing by moving and manipulating objects graphically on screen.

Get a Helping Hand. C++ Designer makes developing in C++ as easy as possible. By generating framework code in C++, it does the hard part for you. What's more – it will help you make the most of your existing investment in C++ Tools – especially Borland's IDE

Implement the Design Factor. C++ Designer is developed around a powerful Object Oriented design methodology – Rumbaugh's Object Modelling Technique (OMT). Rule and consistency checking are built-in, making C++ Designer an extremely effective aid to good design.

Simplicity is the Key to ensuring that tools are used effectively. With C++ Designer it's easy to pick up and get started. An intuitive user interface, on-line help and tutorial ensure that users get that maximum possible benefit right from the start.

The Choice is Clear! At the remarkable price of £195 per copy, C++ Designer will be an essential part of every serious C++ developers toolkit. Call Julia Hennessey TODAY on 0242 226553 for a FREE information Pack.

C++ designer

New release with full Visual C++ Integration - including MFC2 library NOW AVAILABLE. Special Limited Offer only £99.

SELECT
Software Tools

Select Software Tools Ltd, Idsall House, High Street,
Prestbury, Cheltenham, Gloucestershire GL52 3AY
Telephone: 0242 226553, Fax: 0242 251491

> CIRCLE NO. 165

C++ Designer is a trademark of Select Software Tools Ltd.

small 'L' at the start of the name to remind me that this is a procedure returning a logical value. Its code is given in Figure 6.

A TCursor is used to access the USERS table. TCursors work fast because they have no Windows display overhead. Note that the method provides a master password (using AddPassword) and removes it after use (RemovePassword). Without these statements, access to the table would be denied and the open would fail.

A search is made on the ID field of the table using the ID passed to the routine. Since the ID is a keyed field this will be a rapid operation (although in fact USERS is not likely to be a large table). If the ID is found, the password is compared, and if found to be correct, the user name is extracted from the table and passed back to the caller (sName is a VAR parameter).

```
method CONFIRM.PushButton( var EventInfo Event )
var
   sID, sName, sPassword string
endvar
sID = ID.Value
sPassword = PASSWORD.Value
if sPassword = "" or sID = "" then
   MsgStop( "Error", "Both ID and Password must be entered" ) else
   PASSWORD.Value = ""  ; keep from prying coders
   if lGoodLogin( sID, sPassword, sName ) then
      FormReturn( sName )
   endif
endif
endmethod
```

Figure 4 - Confirm.PushButton *event*

```
method Cancel.PushButton( Var EventInfo Event )
   FormReturn( "" )
endmethod
```

Figure 5 - Cancel.PushButton *event*

```
Procedure f.lGoodLogin( const sID string, sPassword string,
     Var sName string ) Logical
var
   l_tc TCursor
   l_OK Logical
endvar
l_OK = FALSE
AddPassword( "boss" )    ; allow us to access USERS
if not l_tc.Open( "Users" ) then
   MsgStop( "Error", "Cannot open USERS table" )
else
   if l_tc.Locate( "ID", sID ) then
      if l_tc.Password.Value = sPassword then
         sName = l_tc.Value
         l_OK = TRUE
      else
         MsgStop( "Error", "Invalid Password" |
      endif
   else
      MsgStop( "Error", sID + " not found in USERS table" )
   endif
endif
l_tc.Close()
RemovePassword( "boss" )  ; we have finished USERS table
return( l_OK )
endmethod
```

Figure 6 - The lGoodLogin *Custom Procedure*

```
method PASSWORD.KeyChar( Var EventInfo Event )
var
   c string
endvar
c = EventInfo.Char()
if c = " " then ; spaces we can do without
   DisableDefault
   return
endif
if size( Self.Value ) = MAX_PASSLEN then
   DisableDefault
else
  EventInfo.SetChar( c.upper() )
endif
endmethod
```

```
method ID.KeyChar( Var EventInfo Event )
var
   c string
endvar
c = EventInfo.Char()
if c = " " then ; space
   DisableDefault
   return
endif
if size( Self.Value ) = MAX_IDLEN then
   DisableDefault
else
  EventInfo.SetChar( c.upper() )
endif
endmethod
```

Figure 7 - The Fields' Keychar() *Methods*

```
method F.KeyPhysical( Var EventInfo Event )
if EventInfo.IsPrefilter() then
   if EventInfo.VChar() = "VK_RETURN" then
      DisableDefault     ; throw away the key
      CONFIRM.PushButton()
   else
      if EventInfo.VChar() = "VK_ESCAPE" then
         DisableDefault
         CANCEL.PushButton()
      endif
   endif
endif
endmethod
```

Figure 8 - The Form's KeyPhysical *Method*

The code is presented in Figure 6. As always, it can be improved. Since IDs and passwords are held in the USERS table in upper case, the user has to remember to use upper case also - clearly unacceptable. We can remedy this by attaching some code to the KeyChar() methods of the ID and password fields as shown in Figure 7. This performs three functions:

- it refuses to accepts spaces, so we will not need to trim the fields after entry
- it restricts the fields to a maximum length of MAX_IDLEN (for the ID) or MAX_PASSLEN (for the password). MAX_IDLEN and MAX_PASSLEN are constants attached to the form.
- it converts the incoming characters to upper case.

We are now better protected against the malevolent user who seeks to wreck our application for the fun of it.

The final touch

The above code will provide the basis for a working login system. There are however a few improvements we can add. If the user presses RETURN or ESCAPE at any time, we would like the dialog to pretend that the a button has been pressed. This is easily done, as shown in Figure 8, by code attached to the form's KeyPhysical method. In the P4W event model, keystrokes for any object on the form are sent first to the form ('pre-filtered') and then despatched to the target object. By placing the code at the form level, we code it only once, rather than coding it for each field on the form

Finally, we can improve the user-friendliness of the interface by putting a message on the status line whenever the user moves the mouse into an input field. Figure 9 illustrates how this can be achieved using a combination of MouseEnter and MouseExit.

Calling Password Dialog

Once you have created and tested the above form together with the USERS table, they are available for use with any P4W application. A sample calling

Geoffrey!

Has someone deleted your copy of Underpants?

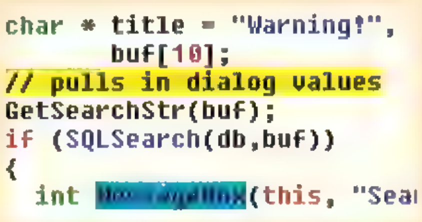
Well, no, I cleared it off my hard disk, actually. Didn't seem any point in keeping it, now that most of my work is in Windows. I use for Windows all the time these days.

Sounds like a rash decision, my son. I bet you'll miss your multiple buffers, configurable keyboard, C-like macro whatsit, undo/redo, multiple compiler support, templates, smart indenting.

This is the 1990s, Brian. Any half-decent programmer's editor does all that stuff. I use ED for the extras: like colour syntax highlighting...

You what?

You Know.



```
char * title = "Warning!",
       buf[10];
// pulls in dialog values
GetSearchStr(buf);
if (SQLSearch(db,buf))
{
  int [illegible](this, "Sea
```

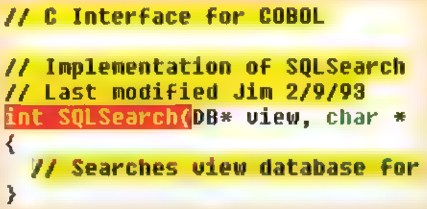
Very pretty. I suppose it does any language, as long as it's C++?

Au Contraire. It'll do any language, as long as it's not APL. And you can define your own keywords. Look: I've got the Windows API set up, plus Jim's COBOL library.

Talking of `SQLSearch(db,buf)`, could you load the source?

I'm not sure, but I think you'll find it in U:\PDS\JIM\UPDATE\LIBSORCE\...

- Strewth! how did you manage that?

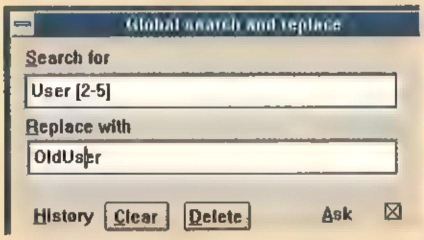


```
// C Interface for COBOL

// Implementation of SQLSearch
// Last modified Jim 2/9/93
int SQLSearch(DB* view, char *
{
  // Searches view database for
}
```

Just right-button clicked on the function name. ED remembers where all your functions live.

Hmm. Still, you can't beat good old grep...

...unless you want to do a multi-file search and replace...



Global search and replace

Search for: User [2-5]

Replace with: OldUser

Sounds more like a jukebox than an editor. How long did it take you to master this ED thing?

That was the best part. ED is a smart editor:smart enough to work the way you do. Which mention of work reminds me: what was it you wanted, Brian?

extensive colour syntax highlighting, smart language sensitive editing for all popular languages, code templates and completion, hypertext function lookup, bracket and object matching, compiler support with error tracking, emulates popular editors, named keyboard macros, regular expression search and replace, search files across all drives and directories (grep), file comparison, unlimited do and redo, context sensitive Windows SDK help, fast C extension language, LAN file-locking, printing with font selection, column and stream blocks, ruler, line drawing, multiple windows + files + buffers, comment alignment, Windows toolbar

Ed for Windows retails at £145. Ed for DOS retails at £99. ED for Windows with ED for DOS costs £199

Don't Hack It - IT

For more information contact:

QBS

QBS SOFTWARE LTD

10 Barley Mow Passage, Chiswick
London W4 4PH, England
Tel: +44-81-994-4842 Fax:+44-81-994-3441
BBS: +44-81-747-1979

► CIRCLE NO. 166

routine is given in Figure 10, although it would be slicker to place the routine in a library containing all the routines for handler user login: `Login`, `Change Password`, `Add New User`, `Delete User` and so on.

Note that while the User Name is returned directly from `wait()`, the User ID is available and can be extracted provided the calling form is aware that the value is stored in a field called 'ID' in the PWORD form. Trying to extract the password does not work because we took precautions... Using fields in this way we can pass variables between forms.

```
   Message( "Enter your Identifier" )
endmethod

method ID.MouseExit( Var EventInfo MouseEvent )
   Message( "" )
endmethod

method PASSWORD.MouseEnter( Var EventInfo
         MouseEvent )
   Message( "Enter your Password" )
endmethod

method PASSWORD.MouseExit( Var EventInfo MouseEvent )
   Message( "" )
endmethod
```

Figure 9 - `MouseEnter` *and* `MouseExit` *events*

```
method AFORM.ABUTTON.PushButton( Var EventInfo Event )
Var
   f form
   sName, sID string
endvar
if f.Open( "PWORD" ) then
   sName = string( f.wait() )
   if sName = "" then
      Message( "Login Cancelled" )
   else
      sID = string( f."ID".Value )
      Message( sName, " logged in, ID ", sID )
   endif
   f.Close()
else
   MsgStop( "Error", "Could not open Password Dialog" )
endif
endmethod
```

Figure 10 - Calling the Password dialog

Conclusion

I hope this article has provided those of you who have already started to grapple with ObjectPAL with some useful techniques you can use. Readers who have not yet had a chance to explore Paradox for Windows should benefit too. Happy Object-ing!

EXE

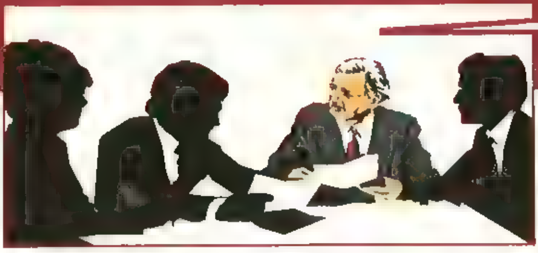
John Braga has worked in IT since 1967 and the strain shows. After 12 years in IBM, working with mainframes, he left in the early days of the microcomputer revolution, to open a Byte Shop in Nottingham. He is now running a small software house specialising in PC database applications, particularly using Paradox for Windows and the Borland Paradox Engine.

He can be contacted on 0480-860349 or on CompuServe as 70134,201.

Nu-Mega TECHNOLOGIES INC

AVOID EMBARRASSMENT!

NEW! VERSION 2.0



"A customer found a bug in our software with a tool called BOUNDS-CHECKER . . . Why aren't *we* using BOUNDS-CHECKER?"

Use BOUNDS-CHECKER™ V2.0 For Windows, The Automatic Bug Finder!

Whether you're the boss, the QA Manager or the programmer, it's your responsibility to ensure that your company's Windows programs are "bug-free" before they get into the hands of customers. If you're developing under C or C++, producing a "bug-free" Windows product is no longer a long and stressful experience.

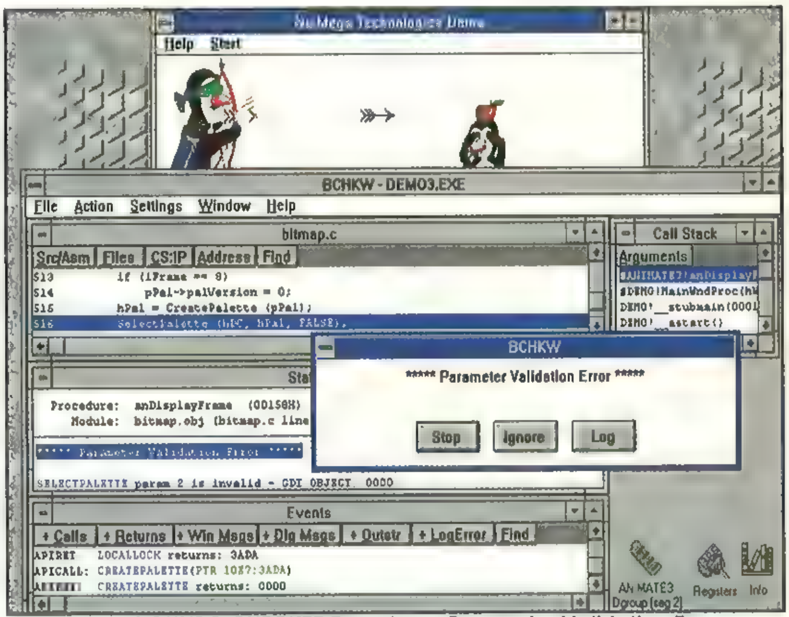
Announcing **BOUNDS-CHECKER V2.0 For Windows**, the software developer's "safety-net". Quickly and easily eliminate the hardest-to-find Windows errors that can take days - even weeks to find like:

- API Parameter Errors
- API Return Value Errors
- Data and Heap Corruption
- Resource Leakage Problems
- Memory Leakage Problems
- Processor Faults

BOUNDS-CHECKER works by transparently setting hundreds of breakpoints within your program to monitor its behavior. When a bug is detected, BOUNDS-CHECKER immediately stops your program and pops up showing the problem. You can then inspect your program's source, variables, stack and heap ... with BOUNDS-CHECKER's powerful display windows.

For those particularly nasty problems, we've introduced new event logging. This lets you look back in time to see what led up to the problem. The events which include messages and API calls can be easily filtered with the click of a button so you can view only the events of interest.

Unlike other debugging tools, there's no learning curve with BOUNDS-CHECKER. Simply select your program's name from BOUNDS-CHECKER's menu; all the rest is automatic. There is nothing to link-in and no macros to compile into your program. All this plus the functionality of a heap checker, debug kernel, API debugger, and a post mortem tool into a single comprehensive automatic bug finder.



Nu-Mega Technologies Demo
Help Start
BCHKW - DEMO3.EXE
File Action Settings Window Help
bitmap.c
Src/Asm Files CS:IP Address Find
Call Stack
Arguments
Sta
BCHKW
Stop
Ignore
Log
Events
+ Calls + Returns + Win Msgs + Dlg Msgs + Outstr + LogError Find
AN MATE3 Dgroup (seg 2)
Registers
Info

```
513         if (lFrame == 8)
514             pPal->palVersion = 0;
515         hPal = CreatePalette (pPal);
516         SelectPalette (hDC, hPal, FALSE),
```

```
$ANIMATE3!anDisplayF
$DEMO!MainWndProc(h
DEMO!__stubmain(0001
DEMO!__astart()
```

```
 Procedure:  anDisplayFrame  (00156H)
    Module:  bitmap.obj (bitmap.c line
***** Parameter Validation Error *****

SELECTPALETTE param 2 is invalid - GDI OBJECT  0000
```

***** Parameter Validation Error *****

```
APIRET   LOCALLOCK returns: 3ADA
APICALL: CREATEPALETTE(PTR 10E7:3ADA)
APIRET   CREATEPALETTE returns: 0000
```

BOUNDS-CHECKER Trapping a Parameter Validation Error

New For Version 2.0

BOUNDS-CHECKER now logs all Windows events. Many Windows bugs are difficult to find because of the event driven, multi-tasking, message based nature of Windows. **BOUNDS-CHECKER V2.0 For Windows** introduces an event logging and display capability that captures all Windows messages, API calls, API returns and other events. The information is displayed in a high-level overview that lets you see how your program is interacting with Windows. When you want to see more detail, simply click to see a section expanded in great detail.

Call now for overnight delivery.

BOUNDS-CHECKER V2.0 For Windows. Only £195 +VAT

BOUNDS-CHECKER For MS-DOS....£159 +VAT

Order both versions & SAVE £105...Only £249 +VAT

BOUNDS-CHECKER, SOFT-ICE, AND NU-MEGA TECHNOLOGIES are trademarks owned by Nu-Mega Technologies, Inc. All other trademarks are owned by their respective owners.

Call: (071) 833 1022
Fax: (071) 837 6411

3-5 Cynthia Street
London N1 9JF

System Science

RISK=NULL
30 DAY
MONEY-BACK GUARANTEE

24 HOUR BBS
603-595-0386

➤ CIRCLE NO. 169

Bringing Virtual Worlds to Life

Sean Ellis describes a language for building virtual worlds that respond to stimuli.

Many Virtual Reality (VR) development systems enable the user to create three dimensional worlds, colour, alter and then explore them. At this stage, the virtual world itself is static and the environment is lifeless. There are no moving objects and no scope for interactivity other than moving the user viewpoint. The world is dead.

However, many applications demand more from the virtual world and an advanced VR development system can bring life to these simulated environments. Thus, objects within a virtual environment can be assigned dynamic attributes and even artificial intelligence. Their characteristics and reactions to various stimuli will mirror movements and activity in the real world. Intelligent virtual worlds are vibrant, dynamic and alive.

Intelligent virtual worlds are vibrant, dynamic and alive

The first programming language specifically designed to control object behaviour in a virtual world is called SCL (for Superscape Control Language). This provides the functionality and flexibility to control simultaneous parallel object activity within a virtual world.

Figure 1 - The BT Network world

In order to allay confusion, it is worth pointing out that the word 'object' in the context of this article, refers to a virtually physical object (a cube or car, for example) rather than a data object as used in OOP.

Virtual World Types

Virtual reality is, by definition, attempting to convey a sense of realism. It is a powerful tool for modelling situations in the real world or making abstract data seem more real. In both cases, realism can be enhanced not only by increasing image quality, but, more importantly, by using motion and complex behaviour on objects.

When classed according to movement and control, virtual worlds fall into four basic areas. First there are applications which present no sense of movement. The static computer-generated worlds produced by architectural CAD programs fall squarely into this category. They may be real time walk-throughs, but there is no interaction; the world is effectively dead.

The second area contains worlds with movement. Most systems provide some form of motion control on objects, from simple dynamics to full simulation of the physical properties of the object in question. Each object reacts in a manner consistent with a set of rules: consider these the laws of physics for that world. This is crucial as a first step toward interactivity. If the user collides with a ball in the virtual world, he would expect the ball to be affected in some way. Using a basic laws-of-physics interaction model, the expected reaction would be for the

Raima Database Engine Captures Fortune 500 With Record Speed
Now Raima Data Manager™
Formerly db_VISTA III

Accelerated Database Performance

Compared to conventional relational databases, retrieval of records can be 10—20—even 50 times faster with Raima Database Manager from Raima Corporation.

Propelling The Biggest Names In Business

Companies like General Motors, Hewlett-Packard, IBM, Eastman Kodak, Rockwell and others are using Raima Database Manager in their competitive environments. Today's most critical, most demanding applications demand the high performance of Raima Database Manager.

Powerfully Efficient Leading-Edge Technology

Raima's combined technology merges the flexibility of relational databases with the lightning speed and efficient storage of the network model. With the program written entirely in C, you can "fine-tune" the Raima Database Manager engine for optimum performance in any application.

Put Yourself In Fast Company

Give yourself the competitive edge of Raima Database Manager:

- Speed—faster access to data
- Portability—supports most environments
- Royalty-free—increase your profits
- Source code availability—total programming flexibility
- Full Raima support services—including training

Whether you're writing a stand-alone DOS application, or one for UNIX accessing thousands of records, Raima Database Manager will put your application on the fast track. Race to the phone and call for more information!

Raima Database Manager™
The High Performance DBMS

In the U.K. call: 44 (276) 685 761

Raima UK Limited,
Lyons House,
2, Station Road,
Frimley, Camberley,
Surrey, GU16 5HF,
United Kingdom.

RAIMA™

Specifications Relational B-tree indexing. Network data model. Relational SQL query and report writer. Single & multi-user. Automatic recovery. Built-in referential integrity. Supports: VMS, QNX, ULTRIX, UNIX System V, Berkeley 4.2, AIX, SunOS, SCO, MS-DOS, MS Windows, and OS/2. Most C Compilers and LANs supported.

Raima Corporation 1605 NW Sammamish Rd. Suite 200, Issaquah, WA 98027 USA **206-557-0200 Fax: 206-557-5200**

➤ CIRCLE NO. 170

ball to move, bouncing off intervening objects, and eventually coming to rest. Extended laws of physics are used in this kind of system for molecular modelling, for example.

The next area is behavioural control. Objects can detect circumstances arising in the virtual world and respond to them in a variety of pre-programmed ways. This can effectively override the laws of physics for the world. Push a car and it will roll in a straight line until stopped. A car with complex behaviour can be made to turn, accelerate and decelerate under its own initiative or make choices about its route depending on the state and position of other objects in the virtual world.

The final area is the so far unrealised goal of true artificial intelligence on objects.

Behavioural Control

The advantages of behavioural control go a lot further than merely providing playthings for the users of largely static virtual worlds. The VR system can be used to envision inherently complex processes, such as the movement of parts through a factory, dynamics of smoke particles and other fluids or for modelling human behaviour.

With active interaction between the user and the virtual world, useful work can be accomplished in a virtual environment. Visualising real world events can be accomplished and actions taken depending on their nature. The objects in the world can be programmed to provide important visual and aural cues for situations requiring immediate attention or for spotting potentially disastrous courses of action.

Many companies are already using behavioural control in their virtual worlds to simulate complex situations for training purposes, effectively using VR systems as low cost simulators.

Type	Size	Description
char	8 bits	integer (character)
short	16 bits	integer
objnum	16 bits	object number
long	32 bits	integer
fixed	32 bits	fixed point real
float	32 bits	floating point real

Figure 2 - Table of basic SCL data types

The Advanced Robotics Research Laboratory (ARRL) at Salford has several industrial robots. These devices are heavy duty and can cause considerable damage if handled incorrectly. By modelling the behaviour of the robot in a virtual world, inexperienced users can be trained in its use without danger to themselves or others. The virtual robot can be programmed by the trainee to accomplish certain tasks in the same way as the real machine.

The virtual world is not insular, however. The user must be included and allowed to interact with it

British Telecom at Martlesham has demonstrated a prototype application using VR for monitoring network integrity and fault finding (see Figure 1). Each node in a large multi-tier network is programmed to route messages around the system, and can be made to fail either randomly or at a signal from the user. The software will then redirect the messages to avoid the failed node, as in the real system. The eventual aim is to use the system as a front-end for remote debugging of the network and recovery of the failed nodes. The behaviour built into each node is far beyond a basic laws-of-physics interaction model.

One particularly striking example of behaviour modelling is VEGAS (Virtual EGress Analysis System), developed by G Keith Still of Colt International to model the way that large numbers of humans behave in emergencies such as a fire. Up to 200 virtual humans each have one of a set of different rules for getting out of buildings. Some will go straight for the exits, others will attempt to stay with their 'children', even if it means going back towards the source of danger, and some may be given the limitations experienced by handicapped people.

Once set up, the simulation is then run and viewed in real time, with the user controlling the viewpoint and any additional hazards that may occur. By modelling a variety of simple behavioural responses to the virtual environment, the actions of a crowd can be predicted with a fair degree of accuracy.

More traditional simulation is being used in military applications for training the operators of guided missile systems. The application is a significant step up from the previous system being used, which involved laser projection of dots of light to denote the target and missile positions. The new system models the flight of the missile according to the inputs it receives from the trainee and presents views of both it and the target in a much more realistic way. This gives the trainee a much better feel for situations that would actually occur in the field, especially effects such as when the missile is obscured by its own smoke trail.

A final, completely different, example is the BBC's Cyberzone programme, where teams of contestants run their virtual counterparts through a maze of puzzles and obstacles. Although a few of the obstacles were controlled by a remote user, all of the puzzles were completely automatic. The behaviour of each puzzle was pre-programmed to interact with the user and react to given responses.

All theses examples run on desktop PCs using the Superscape virtual reality engine in conjection with the Virtual Reality

```
resume(0,1);
if(activate(me,0))
{
  repeat(10)
  {
    moveby(0,100,0,me);
    waitf;
  }
  repeat(10)
  {
    moveby(0,-100,0,me);
    waitf;
  }
}
```

Figure 3 - Example of SCL Code. Object will 'jump' when activated by user

Put an end to software piracy!

Meet the growing family of security keys from Software Security.

Each one a specialist at enforcing your license agreement in virtually any user environment you can think of. Whether it's DOS, UNIX, Macintosh or OS/2... whether it's a single user installation or a LAN.

Simply connect the appropriate key to a single user computer, or a non-dedicated file server in a network, and you control all access to your protected application.

Users, however, won't even know it's there. The keys are transparent and won't impact software functionality or the ability to make back up copies. Normal node and LAN operations are unaffected.

Simple. Unassuming. Ever vigilant. Easy to incorporate into your application package. And quite possibly the most profitable hardware investment a software developer can make.

To find out more, call:

(0784) 430060

THE ULTIMATE PROTECTION FOR SOFTWARE PUBLISHERS

All product names are trademarks or registered trademarks of their respective holders.

Software Security International Ltd

21a The Precinct, High Street, Egham, Surrey, United Kingdom, TW20 9HN

National telephone: Egham (0784) 430060 fax: (0784) 430050 International telephone: +44 784 430060 fax: +44 784 430050

➤ CIRCLE NO. 171

Supports Multiple Methodologies

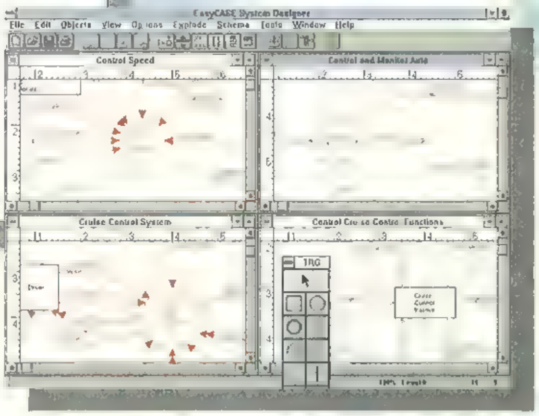
- Yourdon/ De Marco/ Constantine
- Ward-Mellor/ Hatley
- SSADM
- Gane & Sarson
- Shlaer-Mellor
- Martin/ Jackson
- Chen/ Merise/ IDEF1X
- Bachman

Introducing EasyCASE® 4.0

- Provides Full CASE Tool Capabilities
- Powerful Analysis and Design Tool
- Easy to Use
- Affordable

Features

- MS-DOS and Windows Versions.
- Object-Oriented Editing.
- Extensive Documentation.
- Rule-based Methodology
- Monitor.
- dBase Compatibility.
- Methodology Guide.
- Instructive Tutorial.


EasyCASE System Designer
File Edit Objects View Options Explode Schema Tools Window Help
Control Speed
Control and Monitor Auto
Cruise Control System
Control Cruise Control Functions
TRG

"EasyCASE's exceptional charting tools and schema-generation capabilities make it a great value."
PC Week, 13th July 1993

GREAT WESTERN INSTRUMENTS LTD

Tel: 0272 860400

Great Western Instruments Ltd
Redwood House Bristol Road Keynsham Bristol BS18 2BB England

© 1993 Great Western Instruments Ltd. All rights reserved. EasyCASE is a registered trademark of Evergreen CASE Tools Inc.

Evergreen CASE Tools

➤ CIRCLE NO. 183

CLIPPER UNDER UNIX

Produce Executable Standalone UNIX code with no Runtime fees

FlagShip is fully data compatible and remains reverse compatible with most other XBase languages, including dBase III+, dBase IV, FoxBase and FoxPro.

FlagShip is fully source code compatible with both Clipper 5.2, 5.0x, & Summer '87 including all commands, functions, objects, code blocks, pre-processor directives and addit features.

FlagShip

ARCHI PELAGO

ARCHIPELAGO SYSTEMS LIMITED
Phone 051 473 0281 Fax 051 473 0130

➤ CIRCLE NO. 172

Software Education Group

Quality IT training at affordable prices !

If you are interested in IT training now or in the future at realistic prices then you should be aware of what the *Software Education Group* can offer. Future courses include:

Course	Days	Nov	Dec	Jan	Feb	Mar
Advanced MS-DOS	2	22		17	21	28
Advanced Windows	2	24		19	23	30
C Primer	1	29		24	28	
C Programming	4	30		25	29	
C++ Programming	4		6	31		7
Windows Development using Visual C++	5		13		7	14
Structured Analysis & Design	4			31	7	14

Course fees £175 per day plus VAT (discounts for multiple bookings)
All courses held in central London.

For bookings and further details on all our courses telephone **(0532) 307791** or write to:

Software Education Group
Freepost LS5568, PO Box MT66, Leeds LS17 7YY

➤ CIRCLE NO. 173

Toolkit. The various behaviours are modelled using the Superscape Control Language (SCL).

SCL - Object Control

Since behaviour control is so important, a usable method for defining and editing behaviour is equally as important. The traditional method is to build in control for objects with the low-level rendering code. Redefining object behaviour requires the recompilation of the whole VR application. This can be difficult to manage, and the object control programs can only be altered by the original programmers of the low level code.

SCL is a higher level language with a C-like syntax and heavy bias towards virtual world manipulation. Object status and relationships can be interrogated, and any attribute of the object modified in real time. Each object may have its own SCL program, effectively running in parallel with those on the other objects. This allows complex emergent behaviour to be modelled easily if a large number of similar units is required.

Each type of object can be programmed as a single entity, then duplicated as required. An SCL program can refer to its own object, allowing the program to be duplicated with the object and still retain its correct context.

```
if(activate(me,1))
{
  /*
   * I've been hit -
   * copy complaint to other object.
   */

  strcpy('Other_object'.String,"Ouch!");
}
```

Figure 4 - Communication using strings

```
objnum Child;

/*
 * Duplicate this object (me).
 */

Child=dup(me);

/*
 * Move it a bit so it's not on top of me.
 */

moveby(random(10000),
       random(10000),
       random(10000),Child);
```

Figure 5 - SCL for a self-duplicating object

The basic aim of SCL is to allow programs to be written to alter any aspect of any of the objects within a virtual world. Thus it has access to their positions, sizes, shapes and all the other attributes that the Superscape system allows objects to possess.

> ***The inclusion of behaviour control in virtual environments is a fundamental part of making them realistic and useful***

Data Types

It supports a number of basic data types shown in Figure 2, most of which will be familiar to C veterans. In addition to these basic types, arrays and pointers are also available. Strings are dealt with as character arrays, as in C.

The distinction between the 16-bit types `short` and `objnum` arises due to the complexities of maintaining the context of SCL programs as objects are created, deleted and edited in the virtual world..

Multitasking

SCL is a co-öperative multi-tasking language. Each object in the world executes its program in turn, handing off control to the next at convenient points. Objects may take as little or as much time as is necessary to complete their task. This eliminates a lot of the headaches in trying to debug many parallel threads in a preëmptive system (ie one where the execution overseer suspends and resumes execution of the programs at arbitrary times). By resuming the SCL program from where it left off, a program may effectively take several frames to complete (see Figure 3).

Objects can interact with and influence each other in several ways. The most physical way of doing so is to detect collisions. If moving objects collide with others, both are alerted to the fact of the collision, and can optionally retrieve the number of the object with which they have collided.

They can also communicate via a message passing system. Objects can send and wait for messages (often for several frames), acting on their contents when received. The messages passed are very simple - a single 32 bit value - and can be traced back to the sender if a reply is required.

Finally, each SCL program has access to the variables of SCL programs running on other objects. If it becomes necessary, large chunks of information can be passed to other objects by directly manipulating their variables (Figure 4).

New Objects from Old

One of the most powerful commands in a virtual object control language is the ability to create new objects. SCL allows creation and deletion of objects, and, perhaps more crucially, duplication.

With the traditional system of object control, this is nothing special. However, an SCL program itself can be an attribute of the object to be duplicated. Self duplicating objects can rapidly take over the system. An example of some SCL code for a self duplicating object can be found in Figure 5.

Duplication can more usefully be extended by making it dependant on certain conditions or by giving the 'children' different attributes to their 'parents'.

The virtual world is not insular, however. The user must be included and allowed to interact with it. Again, SCL has a set of commands to deal with this on several levels.

The simplest is similar to the physical interaction between objects. The user can click (using a mouse) on any object on the screen. This object is then alerted in the same way as for collisions. The `activate` command in Figure 2 is the test for this condition.

Once activated, the object can display information on screen, make noises or react physically in the virtual world. It can ask for more information using a GUI-style dialog box, and then act on that additional information. Communication also extends to the file system, with C-style file operations, and the I/O port system.

Not Just Objects

Despite its main use as an object control language, SCL is also used in other parts of the Superscape system. The functions performed by input devices (eg activating an object on a mouse click) can be redirected to specific SCL routines to perform complex or unorthodox actions.

It is also valuable in generating shapes. Each object is represented on screen by a shape (eg a cube, sphere, etc). These can be built by hand or generated using SCL programs. When these are combined with the animation facilities in Superscape, the results can be disturbingly biological...

SCL encompasses over 500 commands, and is compiled to intermediate code for speed and space. The source code for SCL is not stored with the objects or indeed at all; only the user's comments and variable names are stored directly. When edited, the source code is decompiled from the intermediate compiled code and the comments and names re-introduced. This ensures a compact final code which is not dependant on any particular machine or processor.

The advantages of using a separate, high level control language are threefold. Functions are tailored specifically to virtual object manipulation, definition and editing of control programs is far faster and the control code is entirely decoupled from other lower level functions such as rendering.

Successful bees are the ones that are attracted to the flowers; successful flowers attract the bees

The Future

A lot of research is going on in the various fields of artificial intelligence. It is reasonable to expect that this will influence the future direction of behavioural control in virtual worlds. Evolutionary growth of behaviour patterns is one of the most interesting of these areas. Bryan Salt, one of Dimension's researchers, has built a model of a simple self-evolving ecosystem consisting of bees and flowers (see Figure 6). Successful bees are the ones that are attracted to the flowers; successful flowers attract the bees. Additional factors influence the bees' choice of flowers, when to breed, how much time is spent foraging, how much looking for a mate and so on. The descendants of both bees and flowers differ slightly from their parents, allowing natural selection to take place. All this is modelled using SCL.

By allowing complex behaviours to 'grow', rather than building them, much more complex interactions can be developed in less time. The evolutionary process supplies the intelligence, rather than having to rely on the programming skill of the virtual world designer. The utility of this approach will continue to grow as more research is done.

Conclusion

The inclusion of behaviour control in virtual environments is a fundamental part of making them realistic and useful. Since behaviour control is so important, a usable method for defining and editing behaviour is also important. Separating the object control from the 3D routines and other parts of the code makes the implementation of object behaviour much easier. The behaviour control is thus decoupled from lower level processes.

SCL is Dimension's solution addressing these issues. It is believed to be the first language specifically tailored for object behaviour control in a virtual world.

EXE

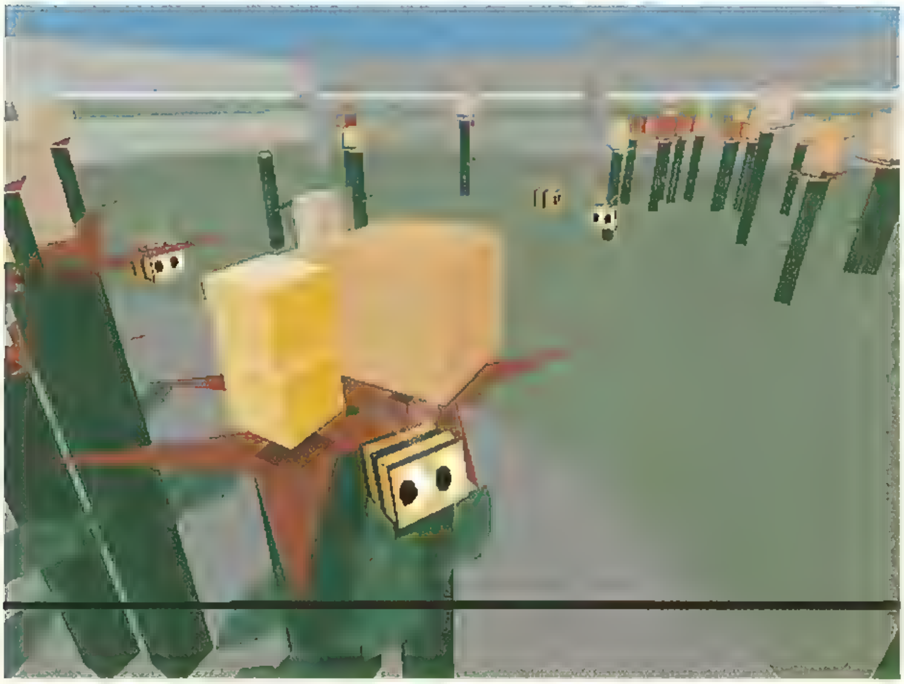
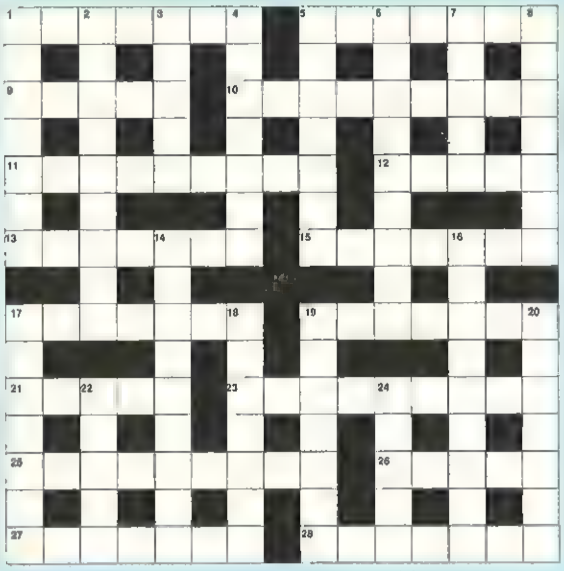


Figure 6 - Bees and flowers ecosystem

Sean Ellis works as a programmer for Dimension International, and is closely involved with the design and implementation of the SCL language.

The Superscape VRT system is available from Dimension International (0734 810077).

NOVEMBER .EXEWORD



ACROSS

1 Screen part of the OS (7)
5 18 of chips? Sounds like a stupid trick!
9 Lamont's standards of 17 ac? (5)
10 Getting rid of the software glitches (9)
11 Savannah is pot country (9)
12 Rossum had the first automated machine (5)
13 Input/output handling routines at the golf club (7)
15 There are 2 in a 26 (7)
17 State how many years old - how mean!
19 Spotty pro's boss went in front (7)
21 Record graphic from cpa program maybe (5)
23 Choose Raegan's particles on which our work is based (9)
25 Carry out design or tool (9)
26 Old byte - piece of music for such (5)
27 Despite the odds, smoothes the problem? (5,2)
28 Lots of 18s neatly arrayed (5,2)

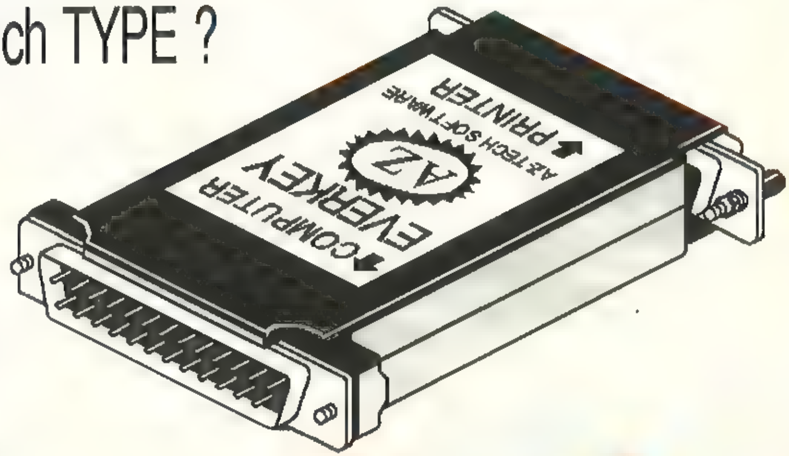
.EXEword compiled by Eric Deeson

DOWN

1 Fellow of some years looked after the system (7)
2 It's natural to bring something to a standard (9)
3 A modern micro can do several at once (5)
4 Circular measures (7)
5 Check syntax of U-boat on returning tide (7)
6 Function to record a repeated beat of sound (9)
7 Many arm, say, and rise up (5)
8 Cancels the Tyneside terminals (7)
14 Precision of former play on loch (9)
16 Everyone shod as no terminal failed to start (3,6)
17 Store for posterity school micro and apiary (7)
18 Entity in an array like Mendeleev's (7)
19 Interactive videotex with remainder in pixel (7)
20 Gem's metaphor (gem of a metaphor) (7)
22 Micro for the teacher (5)
24 Th' runish character is a pain when in the flesh (5)

E	R	R	O	R	■	E	A	R	T	H	■	C	A	R
R	■	O	■	O	■	N	■	O	■	O	■	H	■	A
R	E	B	O	U	N	D	■	B	O	O	L	E	A	N
■	■	O	■	L	■	L	■	O	■	T	■	C	■	G
W	A	T	T	A	G	E	■	T	R	I	C	K	L	E
I	■	I	■	D	■	S	■	■	■	N	■	E	■	■
C	A	C	H	E	■	S	T	A	G	G	E	R	E	D
K	■	■	■	■	■	L	■	N	■	■	■	■	■	R
S	P	I	R	O	G	Y	R	A	■	S	E	P	I	A
■	■	N	■	P	■	■	■	R	■	C	■	E	■	W
L	I	Q	U	E	U	R	■	C	H	A	N	N	E	L
O	■	U	■	R	■	E	■	H	■	L	■	D	■	■
G	R	I	M	A	C	E	■	I	M	P	L	A	N	T
I	■	R	■	T	■	D	■	S	■	E	■	N	■	E
C	U	E	■	E	G	Y	P	T	■	L	A	T	E	X

OCTOBER .EXEWORD



Ticklish X windows

Peter Collinson gives a guided tour of Tcl/Tk, a scripting language for writing X-Windows applets.

Since the beginning, UNIX has provided the user with the ability to create his own commands easily. Okay, you can write a C program to do some jobs, but I am thinking here of generating shell scripts that tailor the environment to make it easier to use. For example, you can create a file called `match` in your private `bin` directory that contains:

```
#!/bin/sh
exec grep -i "$@"
```

After that you wave the magic `chmod` wand:

```
% chmod +x match
```

you end up with a new command called `match` that does a case-independent `grep`. Although you would probably do such a thing with an alias in your shell.

With very little effort, you can create new commands that do what *you* want. Creating this script is massively easier than writing a C program to do the job. These days I resort to writing C programs only when the shell scripts become too large or incomprehensible. I think that there are at least three separate home-grown packages on my system that have shell script front-ends driving specialised C programs that do things such as maintain databases or print text.

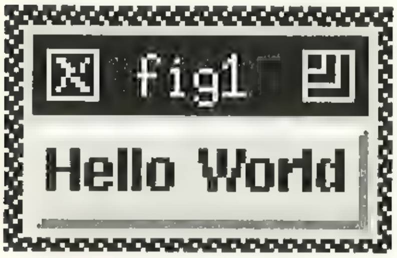


Figure 1 - The Hello World Application

Then along came X. Most software developers probably use X on their workstation as a way of displaying text. Your screen is full of `xterm` programs emulating a text-based terminal. Although the commands in these emulations are configurable, X is not. It's hard to create X programs; it's hard to make personal graphical commands. X has become WYGIWYG (wiggy-wig): what you're given is what you get.

The idea was to provide a single language that could be plumbed into different applications making them extensible

Hopefully Tcl/Tk will change this for you, as it has for me. Tcl/Tk actually consists of two things. Tcl stands for *Tool Command Language*. The author, John Ousterhout, states that you should pronounce it *tickle*, although I resist this. Tcl offers a 'shell-like' scripting language providing variables, loops and expressions. Tcl is interpreted and is supplied as a set of C procedures that can be embedded in applications. The idea was to provide a single language that could be plumbed into different applications making them extensible in the same way that LISP supports the EMACS editor.

Tk is an extension of Tcl. It's a toolkit for the X window system. It takes its look and feel from the Motif model, but does not use Motif from OSF. Tk extends Tcl by adding a set of commands for building user interfaces. It means that you can create X applications from a simple text script of Tcl commands. The Tcl/Tk script can interface to the system using several methods. For instance, it can deal with files: Tcl provides file handling mechanisms. And you can execute standard UNIX commands, since Tcl allows you to execute commands and obtain their output. In addition, you can write your own C routines and embed them in the Tcl/Tk application.

Tcl

Tcl is a fairly simple language to learn especially if you have written some shell scripts. Statements in the language have the general form of a keyword followed by a number of arguments.

```
expr 67+90
```

will print the number `157` on the standard output of the program. The word `expr` here is a built-in Tcl command that does arithmetic for you. The arguments are legal ANSI C expressions. Commands are written one per line, but can be separated by semicolons:

```
expr 67+90; expr 1<<9
```

will print the result of the last statement on the standard output. If a line is too long, you can make it span several input lines by adding a backslash at the end of the relevant input lines.

Variables are similar to those in shells. You can set a variable to a value using the `set` command:

```
set a 67
```

and invoke it using the familiar dollar syntax:

```
expr $a+90
```

printing `157` again. The `set` command is very simple, it only takes two arguments: the name of the variable and the value to be loaded. So

```
set b $a+90
```

will set `a` to the string '`67+90`'. All variables are text strings and are converted when necessary to the type that is applicable for the action being taken. If you want the expression to be evaluated then you must run the `expr` command:

```
set b [expr $a+90]
```

The square bracket syntax is similar to the back-quote syntax in shells. It tells the interpreter to run the command and return its value. Here the value will be loaded into the variable `b`. Square brackets nest, so complex statements can be created.

Tcl also contains procedures:

```
proc CtoF {f} {
    return [expr 1.8*$f + 32]
}
```

The above converts from Fahrenheit to Centigrade. The word `proc` is a statement with three arguments: the name of the procedure, a list of arguments and a procedure body. Both the argument list and the procedure body are contained between braces `{...}`.

The use of braces is perhaps the most elegant thing in the whole language. Braces are used around strings to ensure that their contents are passed into the relevant command without any substitutions being made (like quoting with single quote characters in shells). The most important use of braces is shown by the `proc` above, they are used to defer evaluation. The body of the procedure is not evaluated when the procedure is defined, instead it is stored and evaluated when the procedure is called.

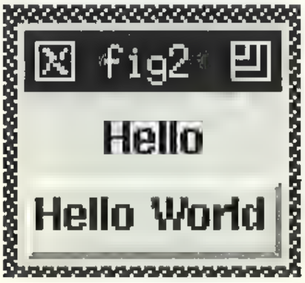


Figure 2 - Adding a label

You can also see this at work in control structures such as `while`:

```
while {$i > 0} {
    puts stdout $i
    set i [expr $i-1]
}
```

The `while` statement has two arguments: a conditional and some code to be executed in the loop. The body of the `while` loop is enclosed in braces so that it is evaluated each time around the loop, rather than at definition time. When you first start using Tcl, you feel that braces are overloaded. However they aren't: they provide a very consistent method of defining data and program segments.

The language has a reasonably full set of control statements. Apart from `while`, there are `if` statements, two flavours of the `for` statement, and a `switch` statement allowing multi-way tests with pattern matching.

The `exec` statement allows you to call a UNIX command and capture its output.

```
set dir [exec ls}
```

The above script will execute the `ls` command in the current directory and place its output in the variable `dir`.

In addition to scalar variables Tcl supports arrays. These are indexed by a name which could be as simple as the number one or could be a more complex text string such as the name of a month. This provides 'associative' arrays like `awk`. Data can also be managed in lists. A list is a number of items separated by spaces or tabs. There are several built-in routines for dealing with this type of data, for example `lindex` will pick out a single member of a list:

```
set v [lindex {a b c d} 3]
```

This will set `v` to the fourth element of the list, namely `d`. Note that indexes start at zero.

Data can also be regarded as strings. Again, there are a large set of built-in operations which can be used to manipulate them. In addition to sub-string and concatenation operations, Tcl has regular expression routines allowing you to search and replace data in strings.

```
set pa [pwd]
regsub -all / $pa " " ps
set first [lindex $ps 0]
```

This sets `pa` to the current working directory by running the built-in `pwd` command. The `regsub` command then replaces all the slashes in the path name with a space and puts the result into the `ps` variable. The `regsub` statement returns zero if no change has been made and one otherwise. (I am throwing this away in the example.) The last statement pulls the first element of the path name into the variable `first`. The double quoted string behaves much like the equivalent shell construct. It preserves spaces and tabs but permits variable replacement. I could have written `{ }` instead of `" "`.

I think that I have given you enough to get a flavour of the Tcl language. Experimentation is helped because the release comes with a program called `tclsh`. You can type text into it like a shell and see what happens. The program is also used to execute scripts. You bung all the statements into a file and say:

```
$ tclsh -f tclsrc
```

Alternatively you can add

```
#!tclsh -f
```

as the first line of the script, turn on the execute bits with `chmod` and have a command written in Tcl. This assumes that your UNIX understands the '#!' convention.

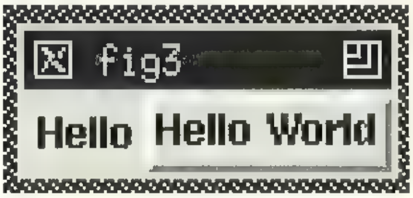


Figure 3 - Changing the packing parameters

Tk

Tk comes with a similar 'shell' called `wish`. If you type

```
$ wish
```

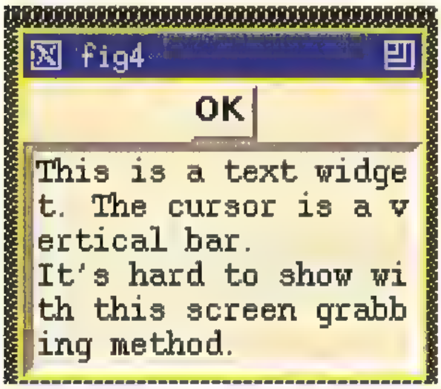
an empty window appears on your screen. You can then type commands to make things happen inside this window. Alternatively you can also type commands into a file and use it as a script.

Tk adds a number of new commands to Tcl giving Motif-like functionality. The basic building block for graphical interfaces for X is the *widget*, a screen object which possesses an appearance and a behaviour. There are several different types of widget in Tk: labels, push buttons, scroll-bars, text areas, sliders, menus *etc*. Widgets have names so that you can refer to them. The names are hierarchical allowing you to address widgets inside widgets for instance. Widgets are named by text strings separated by '.' (as usual pronounced 'dot'). The outer widget is addressed by a single dot, so widget names always start with a dot.

Now we've got enough for an example. The infamous 'hello world' program is:

```
#!wish -f
button .b \
    -text "Hello World" \
    -command exit
pack .b
```

You can see this in Figure 1 (the top line and the outer frame is added by my X window manager `twm`). The `button` command sets up the internal mechanisms and constructs the image of a push button in a widget called `.b`. Like all the widget commands, `button` has a number of optional parameters that are keyword/value pairs with the keyword preceded by a minus sign.



```
This is a text widge
t. The cursor is a v
ertical bar.
It's hard to show wi
th this screen grabb
ing method.
```

Figure 4 - The text widget and exit button

The `-text` parameter tells Tk to load the text string that follows into the visible area of the widget. You can load bitmaps instead of a string if you wish.

Tcl is a fairly simple language to learn especially if you have written some shell scripts

The `-command` keyword is followed by an action that is executed when the button is pressed with the mouse. The effects are 3d and the button does appear to be pushed in when you click with the mouse in the button. In the example, the command is a built-in statement, but you can replace this by any Tcl statement. I usually call a routine here to make something happen.

The `pack` command causes the button to be displayed. Creating a widget does not make it appear on the screen. Instead there are a couple of commands called 'geometry managers' that are responsible for arranging the placement of widgets. It is this notion that makes Tk easy to use.

Buttons

There are various flavours of buttons that are available. Check-buttons allow the user to turn a switch on or off, placing a result in a variable for later testing. Radio-buttons allow you to establish a group of buttons where only one can be selected at any one time. Pressing one will turn any of the others off.

If we want to add an internal title to the existing `hello` program, all that's required is the addition of a couple more lines:

```
#!wish -f
label .l -text Hello
pack .l
button .b \
    -text "Hello World" \
    -command exit
pack .b
```

The first line here defines a `label` widget, which are used simply to display text on the screen. The text to be displayed is the word `'Hello'`. This is loaded into the final image by the `pack` statement as before. You can see the result in Figure 2. Notice how the text in the label is centred in the final frame.

We can change the way that `pack` does its job. By default it packs 'down' the screen by aligning the tops of the widgets. If we wanted to make the widgets appear across the screen then we can tell the first `pack` command to align with the left-hand side of its parent. Change the label `pack` command to:

```
pack .l -side left
```

You can see the result in Figure 3.

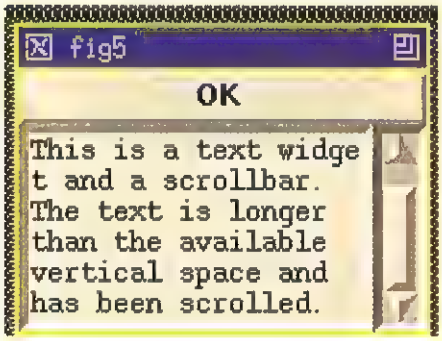
Once you have established a widget, you can apply further commands to it by using its name as a command. For example, I can make the `OK` button inactive by saying:

```
.b -state disabled
```

The button will no longer respond to key presses. The text in the button will be 'greyed' to tell the user that the button is no longer active.

Text entry

There are a few widgets that allow you to type text into the system. The en-



```
This is a text widge
t and a scrollbar.
The text is longer
than the available
vertical space and
has been scrolled.
```

Figure 5 - Adding a scrollbar

Half the cost NOT half the dongle

Softlok II Keys, developed by Softlok International, have been designed to be an effective yet low cost solution to software piracy. Designed and manufactured in the UK for software developers worldwide, Softlok II contains all the flexibility you will need to protect your software.

Softlok II units are sophisticated keys containing 240 bytes of secure non volatile memory. An 8 byte password provides read/write protection. Both the memory area and password can be changed using the supplied routines from your application. As a programming adaptor is not required, they can also be altered in the field.

Our low cost starter pack contains everything you need to get you quickly up and running with routines provided on the disk for most DOS and Windows compilers. We can usually ship SoftlokII Keys for next day delivery so that you can get your software on the market just as quick.

Low Cost (less than half the cost of some dongles)

Starter Kit	£20.00
1-19 Units	£16.90
20-49 Units	£15.40
50+ Units	£13.70

All prices exclude VAT and delivery

Softlok International Ltd
Imex House
40 Princess Street
Manchester
M1 6DE
England
Tel : 061 228 7379
Fax : 061 236 6890

Softlok International Ltd
Nieuwezijds
Voorburgwal 353
1012 RM Amsterdam
Netherlands
Tel : 020 638 3126
Fax : 020 638 3171

FAST

> CIRCLE NO. 175

try widget supplies you with a one line input area. The more generalised text widget provides you with an area on the screen into which you can type. It permits editing with the cursor and keyboard. Here's a small program that establishes an exit button and a text widget.

```
#!wish -f
button .ok -text OK \
    -command exit
pack .ok
text .txt \
    -width 20 -height 6 \
    -relief sunken \
    -borderwidth 4
pack .txt
focus .txt
```

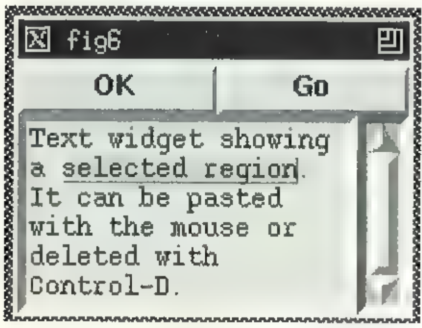
You can see this in Figure 4. I'd prefer the OK button to span the width of the frame so I'll add

```
-fill x
```

to it's pack command on the next iteration. I have told the text widget that I would like it to look as if it was below the surface level of the screen with the -relief option and have emphasised this by providing a four pixel-wide border. Other relief options are flat and raised.

The focus command allows you to dictate the area for input from the keyboard when the mouse enters the area of screen occupied by the Tk application. It's not strictly needed here, but I put it in so I could discuss it. It's useful when you have several input areas and want to switch between them.

I have given the text widget the area I want it to display: six lines of 20 characters. However, the user is bound to need more text, so I have installed a scroll-bar to move the text up and down.



Text widget showing a selected region. It can be pasted with the mouse or deleted with Control-D.

Figure 6 - The 'completed' application

```
#!wish -f
button .ok -text OK \
    -command exit
pack .ok -fill x
scrollbar .scr \
    -relief sunken \
    -borderwidth 4 \
    -command ".txt yview"
pack .scr \
    -side right -fill y
text .txt \
    -width 20 -height 6 \
    -relief sunken \
    -borderwidth 4 \
    -yscrollcommand ".scr set"
pack .txt
focus .txt
```

This is shown in Figure 5. The OK button is packed first, and spans the outer frame. I have then added a scrollbar widget. Its -command code is linked to the text widget. When the scrollbar is altered, the command will be called with an additional parameter giving the amount of scroll that is requested. The scrollbar is packed on the right of the frame and it's told to fill the available space in the *y* direction.

The only change to the invocation of the text widget is the addition of the -yscrollcommand option. When the text widget automatically scrolls up, it will set a new value into the scrollbar. It's also easy to add a scrollbar for the *x* direction too.

Things are beginning to look reasonable. Now let's add a button that takes the user's text and does something with it. We could simply add another button under the OK button, but that would use a lot of screen real-estate. I'd prefer to add it next to the OK button. This complicates the geometry. We need to introduce a container to split the screen area in the middle. Tk provides a widget called a frame that can be used to contain other widgets. We replace the two lines that create the OK button by:

```
frame .f
pack .f -fill x
button .f.ok -text OK \
    -command exit
button .f.go -text Go \
    -command goforit
pack .f.ok -side left \
    -fill x -expand yes
pack .f.go -side left \
    -fill x -expand yes
```

Tk will pack the f frame across the top of the window, and place the two buttons inside it. They are expanded to fill the frame by the '-expand yes argument. Figure 6 shows you the result.

We now need to write a procedure called goforit that does the work. It's easy to pull the data from a text widget.

```
proc goforit {} {
    set s [.txt get 1.0 end]
    ....
}
```

The above procedure sets the variable s to the value of the data inside the text widget from line 1, col 0 to the end. In passing, notice that is how a procedure with no parameters is defined, the { } is a null list.

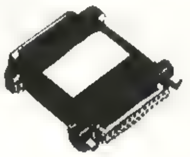
Tk allows you to design the screen image first and then add the pieces that do the work. It's easy to grow applications from an image to a working script. I think it's great.

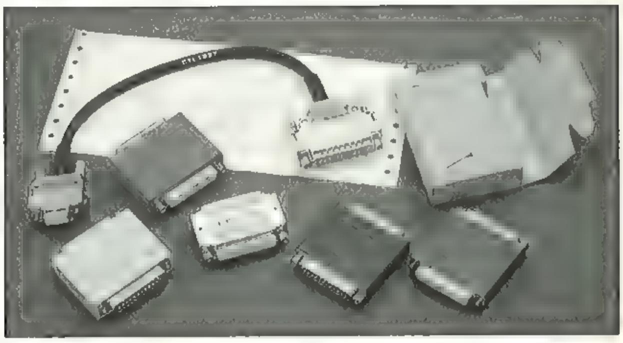
Where to get things

Tcl and Tk are available for public FTP from sprite.berkeley.edu. Many other sites mirror this. For example, the releases are available on the UKUUG archive on src.doc.ic.ac.uk in the packages/tcl directory. Tcl V7.0 is the latest release. You'll find it in a file tcl7.0.tar.Z. The current release of Tk is Tk3.3 which can be found in a file called tk3.3.tar.Z.

While you're there get the files book.p1.ps.Z, book.p2.ps.Z, book.p3.ps.Z and book.p4.ps.Z. These four sections are the PostScript drafts for a forthcoming Addison-Wesley book on Tcl/Tk by John Ousterhout. The releases come with copious manual pages, but this text describes how to use the system.

Peter Collinson is a freelance consultant specialising in UNIX. He can be reached electronically as pc@hillside.co.uk *(although your mailer might be happier to put the address the other way round) or by phone on 0227 761824.*





Further Thoughts on Pointers & Arrays

Part 2: A C++ Perspective

Last month I directed your thinking to some aspects of C's implementation of arrays via mechanisms relying on pointers. I hope that most of you realise how great a burden this places on the programmer. Almost all warnings about pointers need to be taken seriously as they are usually caused by program errors, even if they are not language errors.

Of course you can always shoot yourself in the foot by using a cast. Even when you are dealing with cv (`const` and/or `volatile`) qualified objects you will not get into danger as long as you avoid using casts to silence the compiler messages. The problems start when you attempt stretching to more complicated objects. Arrays of pointers (you will remember that we need these for dynamic arrays in C) have some real potential for trouble, particularly when good programmers try to cover all the bases. Consider the following:

```
char *const *const
  make_tri_array(int size)
{
  char * * ar=
    new (char *)[size];
  for (int i=0; i<size; i++)
  {
    ar[i]=new char[i+1];
  }
  return ar;
}
```

Is the conversion from `char **` to `char * const * const` safe? Legal now? Legal next year? If it is what about other combinations of cv qualified pointers to pointers... If you are certain you know the answers to these questions you should join the working parties standardising C++ because there are some difficult problems hidden in such conversions (and made even worse when we are using a return by value (which the above is - since it is returning the value of a pointer to an array of pointers).

You have been warned. The treatment of cv qualified pointer conversions contains many subtle traps. There is no point in appealing to your compiler to discover what it will accept because all compilers currently on the market either play very safe by prohibiting all cv qualified pointer conversions or allow unsafe conversions. To make matters worse, some of the rules are still under discussion. If you think this is bad, wait until you see some code written by programmers who have discovered that you can pass arrays by reference as long as you remember the priority of the different operators. For example this function that catches an array by reference:

```
void fn(char (& c)[])
{
  cout<<c;
}
```

produces the expected result from:

```
main()
{
  char c[]="Hello World";
  fn(c);
}
```

But try redefining fn( ) as:

```
void fn(char * (& c)[])
{
  cout<<c;
}
```

My compiler starts mentioning the creation of temporaries. Nice of it to let me know, but are you sure that temporaries have not been created for the previous coding also? (As an aside, I wonder what a temporary for a `volatile` object would be like? As `volatile` is often used to deal with memory mapped input, a temporary for such an item would be nonsense, but do the compilers know this?)

From my current understanding of C++, I suggest the following guide-lines for handling arrays:

- Avoid multidimensional arrays in plain old fashioned C style.
- If you ignore the first guide-line, avoid all cv qualification and be very careful of warnings about temporaries.
- Do not try to pass arrays by reference. You are unlikely to know what is hap.ening.
- Be careful of reusing C code handling arrays in a C++ environment. If you do so, expect to have to tweak your code considerably to improve its type safety. A C++ programmer is much better advised to encapsulate all array type objects into classes. This allows the mechanisms of the language to be used in the ways that they were designed to work. The above example of a triangular array is much better dealt with in Figure 1.

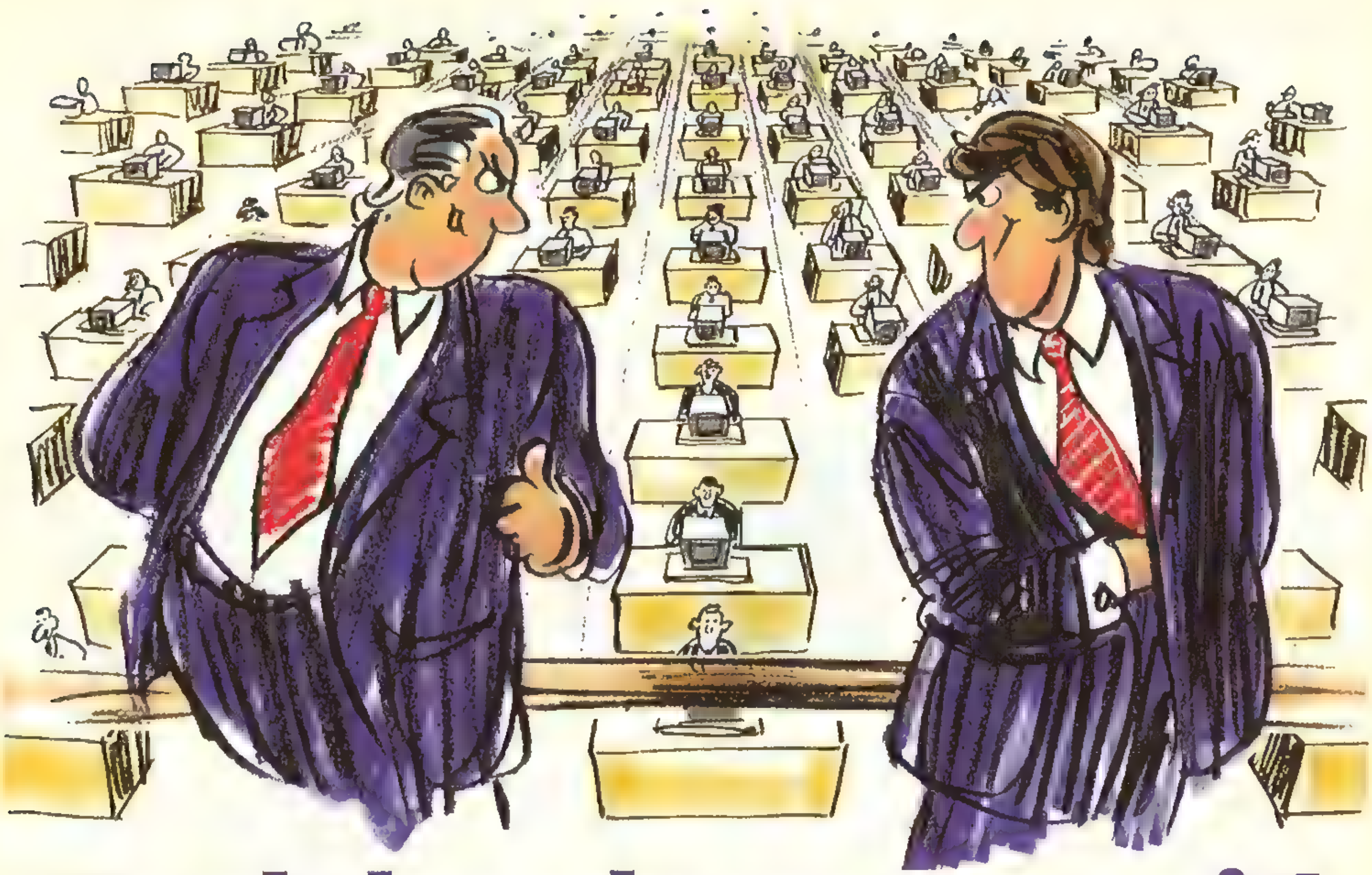
Though I have omitted all the functionality for these classes, I have included rather more source code than I usually do because you will need to look carefully at the code, enter it with your editor of choice and then experiment with it. When you do so, you may discover that some apparently harmless changes produce quite unexpected results. Some will be unexpected because you have not grasped all the implications of C++ and others will be because your compiler is not up to the task, but it is not always easy to determine which is which.

```
class Linear_array
{
  char * const c;
public:
  Linear_array(int n): c(new char[n]) {}
  ~Linear_array(){ delete c;}
};

class Triangular_array
{
  const int size;
  Linear_array ** const la;
public:
  Triangular_array(int n)
    : la(new Linear_array *[n],size(n)
  {
    for (int i=0;
                    i<n;
                     i++)
    {
      la[i]=new Linear_array(i+1);
    }
  }

  ~Triangular_array()
  {
    for (int i=0;
         i<n;
         i++){ delete s[i];}
  }
};
```

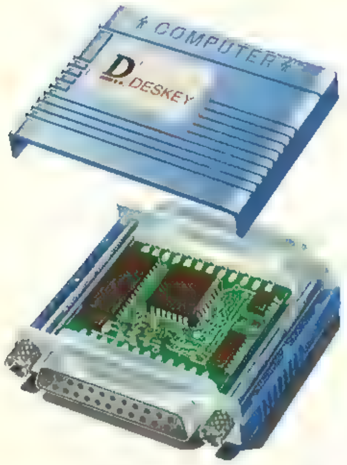
Figure 1 - C++ array handling

EXE

Subscriptions: individual £12, student £6, corporate £75, Overload & C++ SIG £15 (+membership). For further information about ACCU write to Francis Glassborow, 64 Southfield Road, Oxford, OX4 1PA, ring 0865 246490 or email francis@robinton.demon.co.uk.



"I only bought one copy – it's not protected with DESkey"



For years, the illicit use of software has been a major problem. It has been estimated that, in the professional environment, 55% of senior managers have at some time made an illegal copy of software. The solution to this problem is DESkey.

Designed and produced in the UK by our own engineers, with more than 14 years experience in the design of software protection modules (dongles), our current range are the most secure devices available.

- Customer unique seedable pseudo random number generation.
- High speed through encryption. Can also allow secure data storage.
- Up to 16,000,000 down counter for demonstration or 'pay as you use' purposes.
- Algorithmic functions prevent device driver emulation in multi process environments.
- Field programmability allows device redeployment.

DESlock© for DOS

A one line command encrypts any .EXE or COM file securely and quickly with no alteration to source code required, also supports user definable messages and regular DESkey checks. DESlock is also available for networks.

DESlock© for Windows

A one pass encryption system selected from an icon with no programming involvement required to secure the code. Very fast and very secure. A unique product only available from Data Encryption Systems.

- Available as transparent parallel, serial and bus versions for PC's and an intelligent RS232/423 device for any platform (UNIX, VMS etc).
- Drivers available in most major languages and environments including a DLL for Windows. ANSI C code is available for open systems.
- Full lifetime technical support.
- Free fully working evaluation kits.
- Designed and manufactured in the UK.
- Fast order turn around.

Member of FAST
Promoting the legal use of software

Data Encryption Systems Limited

*Silver Street House, Silver Street,
Taunton, Somerset. TA1 3DL*

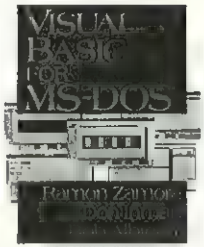
*Telephone 0823 352357
Fax 0823 352358*

➤ CIRCLE NO. 180

Books

A hands-on guide to Microsoft's third-place visual programming language.

Windows Preparation



Basic is very important to Microsoft. It sells more copies than any other language, including C, and it's also set to become the macro language at the heart of all future Microsoft applications, as well as operating systems such as Windows 4.

To help breathe new life into Basic and to adapt it for the Windows environment upon which Microsoft has staked much of its future, it was given a major face-lift and suddenly became visual. No more writing procedural code - now you just draw windows and controls with the mouse and then add the code behind them for the required handling of events.

Having produced Visual Basic for Windows, Microsoft then hit a problem. How do you convince all those Basic programmers using DOS, rather than Windows, to move to a graphical environment? And how do you make it as easy as possible for them to do so? The answer was to develop a version of VB for DOS.

From what Microsoft has been saying in recent months, the future for VBDOS is uncertain. It seems likely that there won't ever be a V2.0 - the product exists purely as a means of making it easy to port between DOS and Windows. This is rather unfortunate, as VBDOS is an incredibly good product for developing DOS applications.

Having said that, there is still the situation whereby Visual Basic, in either flavour, is unlike any other version of Basic that has ever been. Getting one's brain into visual mode is not easy - you have to think events rather than subroutines - and the Microsoft manuals don't do a wonderful job of explaining only the things you need to know and shielding you from the unnecessary-for-beginners stuff.

Which is where *Visual Basic For MS-DOS* comes in. It's divided into 12 main chapters, the first and last dealing with the standard introductions and 'where do you go from here' bits. In between, there are sections on building applications, using sound a graphics, accessing data files, producing menus, and handling errors. Everything that you need to get started, and nothing that you don't.

The chapters on application building take you step-by-step through the creation of a handful of simple programs including a calculator and a rather strange program in which you have to chase a moving object around the screen with the mouse pointer.

The sound and graphics section shows how to produce music and pictures, including a program that displays a piano-like keyboard on the screen which you can play with the mouse. Rolf Harris has a lot to answer for.

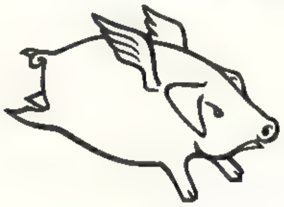
If you're after a definitive VBDOS bible, this isn't it. But if you're after a gentle introduction that's easy to read, *Visual Basic for MS-DOS* is worth a look.

Robert Schifreen

Authors: Ramon Zamora, Don Inman and Bob Albrecht
Title: Visual Basic for MS-DOS *Pages: 310*
Publisher Prentice Hall *Price: £19.95*
ISBN: 0-13-948662-3

Books received this month

Programming on Purpose	P J Plauger	Prentice Hall	£21.25	13-328105-1	pp204
Knowledge Processing and Applied Artificial Intelligence	Soumitra Dutta	Butterworth Heinemann	£19.95	0-750601612-1	pp352
Understanding Relational Databases	Fabian Pascal	John Wiley	£24.50	0-471-58538-6	pp278
Constructing Logic Programs	Jean-Marie Jarquet	John Wiley	£19.95	0-471-93789-4	pp308
C++ Memory Management	Len Dorfman and Marc J Neuberger	McGraw Hill	£26.42	0-83064288-9	pp293
Advanced OS/2 Presentation Manager Programming	T E Burge and Joseph Celi Jr	John Wiley	£29.95	0-47159198-X	pp369
Object Oriented Software Engineering	Steve Halladay and Michael Wiebel	Prentice Hall	£23.95	13-034489-3	pp359
UNIX System V Performance Management	P E Bregman and S A Browning	Prentice Hall	£19.95	13-016429-1	pp359
UNIX System V NFS Administration	Debra Herman	Prentice Hall	£19.95	13-016711-9	pp164
PCTE The Standard to Open Repositories	Lois Wakeman and Jonathan Jowett	Prentice Hall	£22.95	13-06556-X	pp262
HOOD Reference manual 3.1	Dealtte Hertz and J F Muller	Prentice Hall	£22.95	13-396243-1	pp175
An Introduction to Internet Management	Marshall Rose	Prentice Hall	£45.50	13-177254-6	pp456
Easy Object Oriented programming for the MAC using App Maker and Think Pascal	Richard O Parker	Prentice Hall	£31.25	13-092974-3	pp518



SOFTWARE ENGINEER EMBEDDED SYSTEMS

Flying Pig Systems specialises in designing control systems for theatrical and rock+roll stage lighting.

We have an immediate need for a software engineer to work on a new embedded design, using C/C++ on DOS/Windows. The ideal candidate will have 1+ years' relevant experience, and will be able to work in an unstructured environment. Salary £15-£18K.

If successful, you will join a small team working in London and Sydney, but with full responsibility for the software design, coding and testing of a new product.

Please write, with full CV, to:
Tom Thorne, Flying Pig Systems Ltd
53 Northfield Road, London W13 9SY
Fax: 081 579 8469

West Yorkshire and the North

Programmers, Analyst Programmers,
Software Engineers,
Support (Novell, Unix),
also PC Field Service Engineers.

FOR YOUR NEXT CAREER MOVE AROUND WEST YORKSHIRE

Telephone Vincent Atherton on Leeds (0532) 504560 or write to:

AIREDALE RECRUITMENT

Realtex House, Micklefield Lane,
Rawdon, Leeds, LS19 6AX

Airedale Recruitment

the soft corporation

Specialists in Software Development Staff Recruitment

LONDON/HOME COUNTIES WINDOWS SDK/NT DEVELOPMENTS

Senior Development Engineer. **Analyst/Programmer**
To £30,000 + benefits **To £27,000 + benefits**

Strong programming skills in "C" or C++ and Windows 3 are prerequisites for these positions. Experience in some of the following areas is also required: MS-DOS 5.0; MS WINDOWS 3.1; WINDOWS SDK; MS "C" 7.0; VISUAL BASIC 2.0. and Microsoft NT. Also desirable Windows VxD or networking skills. REF SC/01/.EXE

UNIX/X-Windows **ALL LEVELS**

A degree in computer or natural science, 2 years solid C programming experience and a sound understanding of UNIX required to work on large scale programs with user interaction. You will need an intelligent, problem solving approach to work and be a quick learner to programme software in an X-Windows, Windows SDK or NT environment, port software to different UNIX systems and liaise with customers to drive through product improvements. Excellent career opportunities for the right candidate.

£16 - 28,000 REF SC/02/.EXE

C/C++/UNIX/MSDOS

Software Houses and End Users in Finance, Banking, Manufacturing, Commercial, Scientific and Govemment application environments require excellent C skills. Both Windows development skills W/3, SDK, NT, X-Windows and Visual Basic or strong C, C++, solid operating systems and good applications knowledge are again much in demand. Software development experience is the key, and being able to deliver high performance , high quality, well specified software in competitive time scales. Opportunities vary from small to large software companies involved in expert systems, GUI's, Image Processing, GIS, EIS, Communications, Networking and Object-oriented databases. Graduates through to senior software engineers/team Leaders are required. Please call to discuss.

£14,000 - £35,000 REF SC/03/.EXE

RELATIONAL DATABASE SKILLS **ALL LEVELS**

We are currently recruiting software professionals with technical expertise in the following areas: UNIFACE, SYBASE, INGRES (V.5 OR 6), ORACLE V.6,6.3,6.4, INFORMIX, RDB SUPERBASE, REPORT WRITER X-WINDOWS, MOTIF OR OPEN LOOK. Additional experience of: SQL, Forms, C and other 3GL languages are also of interest. We currently have client companies including Management Consultancies, Systems Houses, Systems Vendors, Bank and Finance clients looking for candidates with: Relational Database Design, Database tuning, Systems Administration, DBA's, Pre/Post Sales and solid programming knowledge and expertise. Please call to discuss your particular requirements.

£18,000 - £40,000 + Benefits REF SC/04/.EXE

SOFTWARE ENGINEERS - SENIOR SOFTWARE ENGINEERS

Client End users, Software Vendors and Software Houses dedicated to strategic implementation of leading edge technology and integration of applications across different hardware and operating systems platforms require candidates to degree level with a scientific/technical development bias and 1-3 years experience. There are two main options: TECHNICAL DEVELOPMENT: Continued use of UNIX, VMS, MS/DOS, C, Windows (SDK, NT or X-Windows and Toolkits), Networking and Communications with companies offering technology based careers and management responsibility. COMMERCIAL DEVELOPMENT: Using technical skills already developed, but offering opportunities to apply analysis and design skills rather than remain "a technical guru" in various environments including finance. Please call to discuss your particular career, growth and potential.

£12,000 - £25,000 + benefits. REF SC/05/.EXE

COMMUNICATIONS / NETWORKING

Developments as varied as Voice-activated relational database recognition systems, Protocol enhancement at transport, session and presentation level and World-wide communications systems utilising LAN's and WAN's across different hardware platforms are currently available. Experience of any of the following: ETHERNET, TCP/IP, NFS, X25, X400, X500 in a UNIX, VMS, Sun05, AIX or other flavour of UNIX. Some exposure to structured methods and other leading edge technology , including GUI's and other interfaces, would be a bonus, though training will be given. Knowledge of industry standards and committees is also relevant at more senior levels.

£18,000 - £35,000 REF SC/06/.EXE

UNIX/C or C++ Programmers:

Excellent opportunities exist for bright graduates with one year + experience. Personal background requires a solid understanding of the project life cycle and a commitment to high quality coding. You will be trained in all aspects of Investment Banking, relational databases, 4GL's and Object Oriented Design. A good opportunity for a second career make.

£17 - 25k + Banking Benefits REF SC/07/.EXE

POWERBUILDER **ALL LEVELS**

3 city clients require Powerbuilder skills at any level. Other relevant skills are SQL server, Transact SQL, UNIX, C, C++, Open Client (DB and Net library) and APT.

Exposure to analysis, developing user interfaces and rapid development techniques. Full training in Middle Office/Production systems including: Financial and Management Accounting, Treasury, Equity and Fixed Income.

£20 - 25k + Banking Benefits REF SC/08/.EXE

OOD/OOP, C, C++

As the market for object-oriented skills gathers pace we have a number of clients designing systems in diverse application areas including:

Multi-media, DTP, Telephary, LAN's, Electronic Publishing, On-line Information Feeds, Finance and Banking in both a UNIX and DOS environment.

Positions available very from traditional Programmer/Software Engineer and Analyst/ Programmers to Designers/Senior Software Engineers in the overall strategic direction for end-user organisations.

£17 - 35,000 + Benefits REF SC/09/.EXE

S **the soft corporation**
10 Pakeman Street, London N7 6QN
Tel: 071-609 5501 Fax: 071-700 5787

PROPELLOR HEADS INSTANTLY REQUIRED

- Well, we have three major clients who all seem to be developing similar projects - Financial and Broking Software with C++ and SDK skills with hopefully some knowledge of an SQL RDBMS - although this will be trained up if necessary - seriously outrageous remuneration and benefits.
- MS ACCESS/VISUAL BASIC developers who have a clear understanding of OOP and OOD within a growing Database Consultancy - very hot prospects for the enlightened.
- GUPTA or Sybase or Sybase and GUPTA - Object front-ends? Magnum company wants a major league hotshot development engineer.
- Windows - Powerbuilder - C++ Software Engineering project for small team which is a highly specialist group within a good quality company - good kit - good training and a stable environment - it even comes with a good salary!!
- C Windows Simulation Company with some C++ Development requires an Innovative Developer - very interesting project.

DURNAN LINSTEAD & ASSOCIATES

32 ST JAMES'S STREET
LONDON SW1A 1HT

For any further information and applications, please call Mike Dearing on 071 321 2277 (office), or 081 854 0078 (Evenings/Weekends), or alternatively, fax your C.V. on 071 925 2150

ASH ASSOCIATES

We Specialise in the Recruitment of Software Design Engineers for Real-Time Applications. We have clients throughout the South East seeking engineers with experience of:- Computer Graphics, Image and Signal Processing, Simulation, Virtual Reality, Multi-Media and Interactive Training Systems, Real-Time Control and Instrumentation, Mathematical Modelling, Telecommunications, Data Comms and Network Design.

THIS MONTHS SELECTION OF REAL TIME OPPORTUNITIES

Surrey	Software Engineer, Real Time Control/Imaging C, 68K, Windows	£18K
Surrey	2 x Software Engineers, Telecoms, Real Time Control; C, 68K, Windows	£25K
Hants	Consultant, PC Applications & Databases, C, Windows, Networks	£18K
Oxford	Software Engineer User and PC Interfaces, C, MS-Windows, Z80, 64180	£20K
Southwest	Software Engineer, Math Modelling/Simulation, Fortran, VMS Internals	£20K
Middx	Software Engineer, Real Time Transaction Systems, C, Z80, World Travel	£17K
Middx	2 x Software Engineers, Commercial Control Systems, C, 80X86/96, 68K	£25K
Berks	2 x Software Engineers, Information Systems, C, Windows or Vax/VMS	£20K

WE ALWAYS HAVE A REAL NEED FOR ENGINEERS WITH ANY OF THE FOLLOWING SKILLS

C & C++	MS-WINDOWS	X-WINDOWS	UNIX	MS-DOS	KERNELS	
ASSEMBLERS	COMPUTER GRAPHICS	OSF MOTIF	SIGNAL PROCESSING			
4GLs	TRANSPUTERS	NETWORKS	MULTI-TASKING	TCP/IP	OS 9	
DEVICE DRIVERS	INTERFACING PORTING	METHODOLOGIES YOURDON	X25			
VISUAL BASIC	VISUAL C++	OS 2	IMAGING	ROBOTICS	GUI	VMS
COMPILERS	VME	DATABASES	TELECOMMUNICATIONS	VIDEO	OSI	

Call Ron Cook or James Hunt today to discuss your next move.
Tel: (0425) 475480 or Fax: (0425) 480807 or Post your CV to Ash Associates, Recruitment Consultants, 3 Pipers Ash, Ringwood, Hants BH24 1UF.

ASH associates

LATEST VACANCIES

● NO DINOSAURS!! (C, C++, OOD, UNIX)
Midlands **£17K-£25K**

If you are a Software Engineer with at least 12 months C or C++ experience then this is a brilliant opportunity for you to join a young and expanding Tele-Data Communications company, and work on their range of new SDH projects. You will play an important part in the growth of their design department, joining at an exciting time for you and the company. Ideally you should be Degree qualified and in addition to your C and C++ experience any knowledge of the following would help:- UNIX, OOD, Assembler, CASE Tools, ISDN, SDH.

● FIRST CLASS
£28K-£27K

First Class Honours and/or PhD, with excellent A levels?? Two leading UK and European consultancies are looking for high calibre Engineers to work on some very demanding Scientific and Technological projects. These include NEURAL NETWORKS; GRAPHICS & MULTIMEDIA; COMPUTER SECURITY; ADVANCED COMMUNICATIONS & COMPUTING. Growth is by attracting only the highest quality technical staff and offering them the rewards, challenges and career path that would be difficult to match elsewhere.

● C++, WINDOWS
London **to £20K**

Young PLC specialising in Multimedia have several openings for well qualified (2:1 min) Software Designers. Ideally you will have at least 12 months experience in some of the following:- C++, WINDOWS, C, OOD. It is important that you have the enthusiasm and commitment to work in a small company. We have a full job description and Company information so please call.

● SOFTWARE ENGINEERS
Hants & Shropshire **To £22K**

A division of an international Electronics company specialising in Mobile Communications are urgently seeking several Software Engineers to work on a range of new cellular comms projects. Ideally you should have at least 2 years post graduate experience in C and Assembler for Real Time Control. Excellent facilities and benefits are available for the right applicants.

Contact PAUL SLOUGH or PAUL JONES on 0442 870770 anytime, or write enclosing your CV to the address shown below.

JONES RESOURCING
Audley House
Northbridge Road
Berkhamsted
Herts HP4 1EH

☎ 0442 870770
anytime

MS Windows/C++

Surrey **to £26,000**

This leading developer of graphics based workstations is seeking a high calibre Development Engineer. You need to have achieved a good University degree (2.1 or above) and have two years C, MS-Windows experience. Preferably you will have an exposure to C++ and 80*86 assembler. This is an excellent opportunity to work at the leading edge of technology with full training in Windows NT. Ref: AW4500

Low Level Unix Challenge

Berks **to £27,000**

This leading developer of fault tolerant Unix computer systems concentrating on Porting SVR3 and SVR4, device driver development, kernel hardening/enhancements and I/O controller development is seeking a Unix Kernel Specialist with at least two years kernel/device driver development experience. You will have achieved a good degree and have exposure to C. Future development will lead you to RISC and 88000 development. Ref: AW4501

AI, Windows3, C++

Imaging Start Up

Cambs **£open**

My client requires 2 exceptionally talented software designers to join a team developing novel PC based image processing/inspection systems. The right candidates will have excellent academic qualifications, a minimum of 2 years C++ and Windows 3.1 experience and be capable of holding their own in a very small and high calibre free thinking company environment. One of the most challenging opportunities currently available. Ref: MJ4227

Image Processing

Surrey **to £22,000**

A rapidly expanding order book has moved this leading image processing systems developer into a phase of recruitment. They now seek a number of degree qualified Engineers with a minimum 12 months post graducate software development experience. You will have an in depth knowledge of MSDOS and 80*86 assembler and experience of some of the following; C, OCCAM, Transputer, C++ Unix. Ref: AW4502

Graduates with C++

Surrey **to £20K**

This rapidly expanding software house urgently requires a nuber of graduate Software Engineers who can demonstrate 6 months+ experience of C++ development from either a final year project or work placement. You will be developing datacomms and GUI software so any experience of these would be particularly relevant. Ref: MJ4221

Technically Brilliant?

386/486 Low Level Software

Bucks **£18-£28K**

One of the UK's most successful PC based product developers require exceptionally bright and technically competent engineers to join their development group. Based in a coverted farmhouse, you will have access to whatever development bit you require and be responsible for software design/development from the ground up. Your experience must include a good technical degree (2.1 or betterO) and experince of developing PC based software/hardware products at hardware interface level. Ref: MJ4220

For more information on the above call Mike Jenkins or Adrian Wagstaff **0442 231691** days or 0582 456417 eves/wkends quoting the relevant ref no. Alternatively send your cv to:

Executive Recruitment Services

Hempstead House, Selden Hill, Hemel Hempstead, Herts HP2 4LT or fax cv to 0442 230063.

CATALYST CONSULTING

Search Selection and Advertising
Park House • Greenhill Crescent • Watford Business Park • Herts • WD1 8QU
Tel: 0923 240139 • Fax: 0923 249436

ANALYST PROGRAMMER C and Ingres or OOP Exp 60K

Highly respected international bank require development staff at all levels to develop front and back office systems for their capital markets business areas. They are seeking top calibre developers to contribute to the delivery of an all embracing global system based on state of the art technology.

ANALYST PROGRAMMER C Sybase or OO Exp £30K

Opportunity for bright programmer to break into the prestigious world of high finance. Superb long term prospects.

ANALYST PROGRAMMER £35K + banking benefits

Opportunity for bright developer with Exp in C, C++ and Visual Basic to work on high budget front office application for well respected international bank. Superb prospects.

X-WINDOWS a.p. £30K + benefits

TEAM LEADER £40K

Strong Object Oriented development skills particularly with C++ or Smalltalk, and experience of leading projects are the qualifying skills to take control of development for this exciting financial environment.

WINDOWS SDK, Visual Basic, Sybase C++ and C £20-£40K

Are the main development languages employed by this international conglomerate who are seeking career minded Analyst Programmers to join their ambitious leading edge projects.

VMS, C ANALYST PROGRAMMER £40k + benefits

Top investment bank requires seasoned professionals to deliver risk management systems for their derivitives business.

THESE ARE A SMALL SAMPLE OF OUR CURRENT VACANCIES. IF YOU ARE CURRENTLY LOOKING TO CHANGE JOBS OR ARE CURIOUS ABOUT WHAT OPPORTUNITIES MAY BE AVAILABLE TO YOU, CALL US NOW - ESPECIALLY IF YOUR SKILLS INCLUDE ANY OF THE FOLLOWING:- OO, C++, Windows SDK, X-Windows, Visual Basic, C, Smalltalk,Sybase, Ingres, Oracle, Informix, GUIs, OS/2, Presentation Manager, Motif, CASE tools.

ADVERTISERS INDEX

ADVERTISER	PRODUCT/SERVICE	CIRCLE	PAGE
Admiral Software	Programming Tools	158	61
AP Software	Shareware	148	46
Archipelago	Programming Tools	172	80
Bits per Second	Graphics Tools	157	59
Borland	Development Tools	124	5
Brent Communications	MAX Copy Protection	176	89
Citadel	Comms Library	145	43
Clarion Software	Database	143	39
Computer Associates	Programming Tools	142	37
Contemporary I	BASIC Database	144	41
Contemporary II	Programming Tools	153	51
CTL	Copy Protection Hardware	177	89
DES	Software Protection	180	91
Expotech I	Development Tools	122	IFC
Expotech II - VII	Development Tools	159-164	63-68
Forté Software Tools	Version Control	139	33
Great Western Instruments	Software Design Tools	183	80
Grey Matter	Programming Tools	123	3
Hoskyns	Development Tools	130	17
Hypersoft Europe	Programming Tools	134	24
IBM	OS/2	126	9
Inmark	C++ Application Framework	182	OBC
Intasoft	Version Control	155	53
Kibworth	Training	150	48
LBMS	Seminars	131	19
Magnifeye	Software Protection Device	152	49
MAXPro	Security Products	147	46
Microcosm	Copy Control	168	74
Microsoft Press	Books Specialists	140	33
NeXT	NeXTSTEP	132	21
P&L Systems	NeXTSTEP	133	23
PC Bookshop	Books Specialists	151	48
POET	POET/Object Oriented Database	156	57
Polyhedron	Programming Tools	167	74
QA	Training	181	IBC
QBS	Clipper Tools	166	73
Ralma	Programming Tools	170	77
Rainbow Technologies	Security Products	149	47
Readmar	Version Control	137	29
Real Techniques & Methods	CASE Tools	146	45
Richfords	Training	129	13
Ridgeway Press	Computer Manuals & Printers	179	89
Salford Software	Fortran for DOS & UNIX	154	52
Select Software Tools	C++	165	71
Sequiter	Development Tools	138	31
Softlok	Piracy Protection	175	87
Software Education Group	Training	173	80
Software Security	Security Dongles	171	79
System C	Application Generator	135	25
System Science I	Programming Tools	136	27
System Science II	Programming Tools	169	75
SystemStar SoftTools Ltd I	Programming Tools	125	7
SystemStar SoftTools Ltd II	Programming Tools	127	11
SystemStar SoftTools Ltd III	Programming Tools	128	11
Trevan Design	Bulletin Board	178	89
User Friendly	Software Copy Control	174	83
Zinc	GUI Library	141	35

Letter From Soho

Our vice and broccoli corresponent reports from the new home of .EXE Magazine.

Quarter past nine and the broccoli's a 'pand a pand!' If you don't know this when you arrive at the office, you soon will as Berwick (pr Berrik) Street Market is right outside the window. Berwick Street is London's best known outlet for cheap and cheerful veg.

If you really don't want to know the price, look away now. Oops, sorry, I'm lapsing into Desmond Lynam mode. If you really don't want to know the price of Broccoli, you can always shut the window and roast (a PC with a colour monitor means never having to say: 'would someone **please** turn the heating on - it's bloody November already').

Just after eleven and mushrooms are '80 anarff!!' In those trendy magazines that dealers and VARs read, Soho is a new way of selling PCs. If the corporates have stopped buying truckloads of the things, you can always sell to the guy who wants a cheap box to use at home, so he can knock up CVs without the boss noticing, and run Fight Simulator 5 (you choose whether to play Microsoft or IBM). Small Office Home Office, it stands for, apparently.

Yet here in central London, Soho stands for, well, Soho actually. It probably used to stand for something else, like the Tribeca (pr Try-bekker) region of New York City stands for the triangle below Canal Street. And the broccoli is now a 'pand-a-bagful!!!'

Working in Soho is a strange experience, made even more so if you've moved from the leafy London suburb of Chiswick. Just a dozen or so stops away on the District line, but it might as well be on another planet. In Soho you express yourself by wearing tight leather trousers, no underwear, and a 19 year old blonde. In Chiswick, you ask for your Big Mac without gherkin, if that's okay.

Just to make us feel welcome, the residents of Soho have made sure that we're never far from the cut-and-thrust world of high tech and communications. Opposite the office, above the chip shop, is a walk-in centre where anyone with urgent files to transmit can turn up with a 5 1/4" or 3 1/2" floppy disk. Though 3 1/2" ones are costlier. And 8" floppies are absolutely free.

It's a very strange looking place, this comms centre. The door is always open, for a start. And the modem is a big red box which looks dead old-fashioned to me, but they (the owners) are obviously proud of it as they insist on displaying it in the window. They even illuminate it with neon in the evenings. It has MODEM written in big letters, too, in case you don't know what it is. Actually, it's spelt MODEL. Still, with all these broccoli salesman around it's a wonder it's not MODE'L.

I've mentioned this comms bureau to a couple of friends, but they weren't impressed. They just laughed, and asked me things like:

'Is she a large model or small model?'

'Does she offer an anti-virus service?'

'I suppose she's fully reëntrant.'

'Does she do client/server?'.

Some people just don't take comms seriously. Damn, the Broccoli's all gone. Looks like it's cheese sauce on toast for me tonight.

THE WINDOW LEADERS

Real windows experts learn from QA.

EFFECTIVE USE OF WINDOWS™

For Windows users who want to be expert users; using the tools and techniques described in the course will result in greater productivity in everyday work.
2 Days

SUPPORTING WINDOWS

In-depth coverage of installation, configuration and support topics allows support staff to produce better Windows installations maximising, the return on Windows investment.
4 Days

SUPPORTING MICROSOFT WINDOWS FOR WORKGROUPS

In-depth coverage of installation, customisation and support topics will allow support staff to get the best out of Windows for Workgroups.
2 Days

PROGRAMMING WITH VISUAL BASIC™ PROFESSIONAL EDITION

Comprehensive coverage of the Visual Basic language, capabilities and programming environments, will lead to greater productivity in application development.
4 Days

SUPPORTING WINDOWS NT ADVANCED SERVER

Supporting Windows NT Advanced Server, with extra LAN Manager features, remote access, Macintosh support, and the management of multiple servers. *5 Days*

SUPPORTING WINDOWS NT™

Installation and support of the Windows NT 32-bit multi-tasking operating system and its subsystems such as DOS and 16-bit Windows.
4 Days

MICROSOFT® WINDOWS PROGRAMMING

QA's highly successful and long established Windows programming course provides not just 'how to' information but also an understanding of the underlying principles of how Windows works.
5 Days

MICROSOFT WINDOWS PROGRAMMING WITH C++ AND THE MICROSOFT MFC LIBRARY OR MICROSOFT WINDOWS PROGRAMMING WITH BORLAND C++ AND OWL

For programmers who want skills to write applications using their chosen class library.
5 Days

ADVANCED WINDOWS PROGRAMMING

For established Windows programmers, this course increases knowledge and understanding of Windows, and is invaluable in deciding the best technique to solve a problem.
5 Days

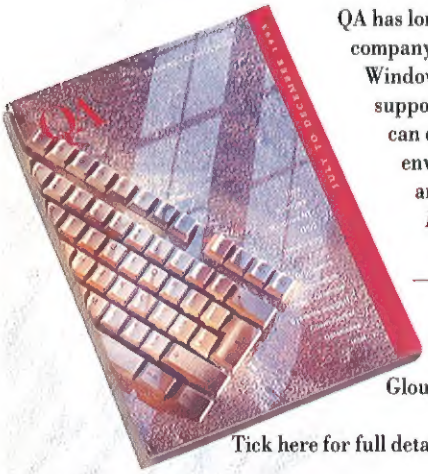
CONVERTING 32-BIT PROGRAMS TO MICROSOFT WINDOWS NT

For developers experienced in 32-bit multi-tasking operating systems, this provides the fastest path to NT development.
2 Days

WIN 32 PROGRAMMING FOR MICROSOFT WINDOWS NT

This course provides 16-bit DOS and Windows programmers with in-depth technical information and teaches good 32-bit program design and application architecture. *5 Days*

Microsoft and MS-DOS are registered trademarks and Windows, Windows NT and Visual Basic are trademarks of Microsoft Corporation.



QA has long been recognised as the leader in Windows training. The world's first company to teach Windows programming. The first to give technical training in Windows NT. Our unique range of courses – Windows training for developers, support personnel and users – are widely acknowledged to be the best. Only QA can offer you the breadth of experience in every aspect of the Windows environment. Experience which is your assurance that QA Windows Training and Consulting will be certain to maximise your skills and effectiveness.
For more information, call Gordon Ritchie on 0285 655888, or complete and return the coupon today.

QA

Mail to QA Training,
Cecily Hill Castle,
Cirencester,
Gloucestershire, GL7 2EF. UK.

Tick here for full details.

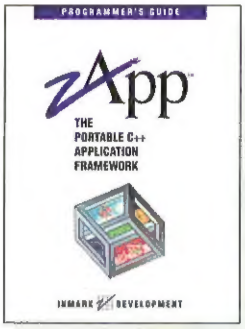
□ TRAINING
□ CONSULTING
□ Windows
□ Unix

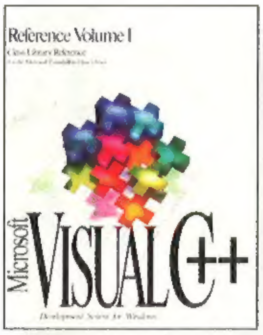
□ OS/2
□ Networks
□ Languages
□ Object-Oriented

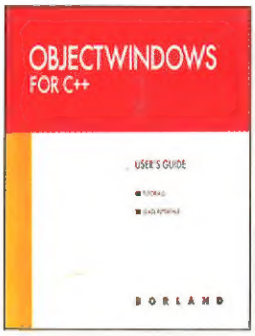
□ Email/Groupware
□ Support
□ Client-Server/Databases
□ Intel

Name ______________ .EXE 11/93
Job Title ______________
Company ______________
Address ______________
____
Postcode ______________
Telephone ______________

➤ CIRCLE NO. 196



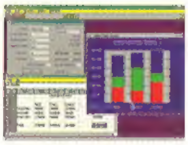




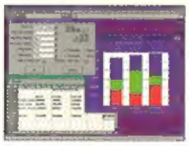
VERY PORTABLE BARELY PALATABLE

WINDOWS

WINDOWS NT


OS 2.x

UNIX X/MOTIF


DOS GRAPHICS

DOS TEXT

Ever get the feeling that OWL stands for "Outdated Windows Library"? That MFC really means "Microsoft Frustration Classes"? If so, toss them out and join the thousands of programmers who have discovered a better application framework: zApp.

In an industry that's always changing, the future belongs to programs that can change with it. That's why we've developed zApp, the industry leading C++ GUI class library. zApp allows you to support all the platforms you need with just one set of source code. You simply make calls to zApp classes, without worrying about the platform's specific API, and zApp does the rest.

zApp is also the most full featured application framework available. Giving you more classes than OWL and MFC combined, zApp sports an unmatched list of capabilities for creating first-class applications quickly and easily. That's one of the reasons why Windows Tech Journal wrote "zApp is simply the best designed application framework on the market. If C++ is your language of choice, then you should be using zApp. Period".

Finally, zApp is easy to use. By providing an intuitive class structure and plenty of example programs, zApp allows you to produce more code faster, making it ideal for C++ programmers of all levels. With over 1200 pages of documentation, including extensive tutorials and demos, as well as free technical support, you know you'll never be left hanging.

So pick up the phone now and call **(0992) 500919** and ask for our free demo disk, and get a glimpse of what the future has to offer.

zApp™

- **Source Code Included**
- **No Runtime Royalties**
- **Compatibility with Most Major Compiler**
- **Documentation** - including over 65 sample programs, tutorials and 880 page Programmers Reference
- **Over 35 Window Classes** – including application, Dialog, MDI, all Standard Controls, Common Dialogs, and access to custom controls
- **Data Entry Classes** - includes Text, Picture String, Numeric, Date, Time, Radio Group, Checkbox and List Fields
- **Graphics classes** – including Windows, Bitmap, Printer, Fonts, Pens, Brushes, Bitmaps, Colors, and Regions, Printer and Font enumeration
- **Object Persistence** – Using a streams interface, easily store objects and other data to disk and retrieve them
- **Advanced MDI Support**
- **Robust and Efficient Memory Management**
- **Advanced Message Handling**
- **Advanced Debugging Support** – traps many common programming errors - includes a powerful assert system
- **Dynamic Sizers** – Sizers provide automated window and graphical object layout management
- **Advanced Scrolling Classes**
- **Menus** - including standard, system, and popup
- **Full DDE Support**
- **Much, much more**

INMARK DEVELOPMENT
2065 Landings Drive, Mountain View, CA 94043
All product names referenced herein are trademarks of their respective companies.

SYSTEMSTAR
SoftTools Limited
1-3 Parliament Square Hertford SG14 1EX
Telephone: (0992) 500919 Facsimile: (0992) 554261

➤ CIRCLE NO. 182